****

# 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** | **4/21/2024** | **April Rose** |  |

## 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

April Rose

## Algorithm Cipher

For the Artemis Financial web application, I recommend deploying the SHA-256 encryption algorithm cipher to enhance security. SHA-256, part of the SHA-2 family, generates a 256-bit hash value from input data, providing a fixed-size output that is computationally infeasible to reverse-engineer. This cryptographic hash function operates by repeatedly hashing input blocks of data, ensuring data integrity and preventing unauthorized tampering. SHA-256's resistance to collision attacks makes it highly improbable for two different inputs to produce the same hash value, ensuring robust data integrity verification. Unlike encryption algorithms, SHA-256 does not involve keys but is commonly used in conjunction with them for data integrity verification, such as in digital signatures. Introduced by the NSA in 2001, SHA-256 has become widely adopted due to its strong security properties and remains a standard choice for applications requiring secure hashing, including digital signatures and TLS protocols. Deploying SHA-256 will effectively safeguard Artemis Financial's web application against security vulnerabilities and ensure the integrity of client data and financial information.

## Certificate Generation

Insert a screenshot below of the CER file.

A black screen with white lines

Description automatically generated

## Deploy Cipher

Insert a screenshot below of the checksum verification.

A screenshot of a computer

Description automatically generated

## Secure Communications

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

A black rectangular object with numbers

Description automatically generated

## Secondary Testing

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

A screen shot of a computer code

Description automatically generated

A screenshot of a computer program

Description automatically generated

[Location](D://Software%20Secrity%20CS-305/Module%207/CS%20305%20Project%20Two%20Code/ssl-server_student/target/dependency-check-report.html)A screenshot of a computer error

Description automatically generated

## Functional Testing

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

A screenshot of a computer screen

Description automatically generated

A screen shot of a computer code

Description automatically generated

A screenshot of a computer program

Description automatically generated

## Summary

Project 2 aimed to enhance the security measures of Artemis Financial's web application by implementing modern encryption technologies and secure communication protocols. The project involved several key tasks:

* Algorithm Cipher Recommendation: After examining the software's vulnerabilities, an appropriate encryption algorithm cipher, SHA-256, was recommended and justified. The choice was based on its financial acceptability and robustness.
* Certificate Generation: Self-signed certificates were generated using Java Keytool in Eclipse. The certificates were exported as CER files, and screenshots of the generated files were included in the report.
* Deploy Cipher: The chosen cryptographic hash algorithm (SHA-256) was deployed by refactoring the code. Functionality was demonstrated with checksum verification, ensuring data integrity during transfers.
* Secure Communications: Refactoring of the code included conversion of HTTP to HTTPS protocol in the application.properties file to ensure secure communication. A screenshot of the secure webpage accessed via HTTPS was provided as evidence of successful implementation.
* Secondary Testing: The refactored code underwent secondary static testing using OWASP Dependency-Check Maven to ensure compliance with software security enhancements. Screenshots of the refactored code executed without errors and the output report from the dependency-check static tester were included.
* Functional Testing: Manual review of the code identified syntactical, logical, and security vulnerabilities. A screenshot of the refactored code executed without errors was included in the report.

Additionally, future recommendations were made, including implementing login verification, double authentication, and ongoing maintenance to keep the application up to date with evolving security standards. The importance of software security in maintaining the reputation and liability of the company was emphasized, highlighting the need for continuous vigilance and updates in response to emerging threats.

## Industry Standard Best Practices

Utilizing industry best practices in software development is important for mitigating cyber threats, data breaches, and security vulnerabilities. Adhering to these practices equips developers with a comprehensive understanding of data privacy regulations, thereby minimizing the risk of adverse impacts such as compromised data and financial liabilities. By implementing best coding practices, developers establish a framework of guidelines, techniques, and procedures within their software applications to fortify defenses against cyber-attacks. These practices, rooted in industry standards, not only bolster software security but also safeguard the reputation and interests of the organizations they serve. Embracing industry best practices in secure coding ensures the resilience and integrity of web applications, providing a robust defense against evolving cyber threats and ensuring the trust of stakeholders in the digital ecosystem.

**Resources**

Oracle, D. (2017). *Java Security Standard Algorithm Names*. Docs Oracle. Retrieved February 17, 2023, from <https://docs.oracle.com/javase/9/docs/specs/security/standard-names.html#cipher-algorithm-names>

ssl, global. (2021, June 14). *Symmetric vs. asymmetric encryption - what are differences?* SSL2BUY Wiki - Get Solution for SSL Certificate Queries. Retrieved February 17, 2023, from <https://www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences#:~:text=Symmetric%20encryption%20uses%20a%20single,and%20decrypt%20messages%20when%20communicating>.