****

# 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** | **2/23/2024** | **Justin Farquhar** | **Updated initial responses to prompts.** |

## 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

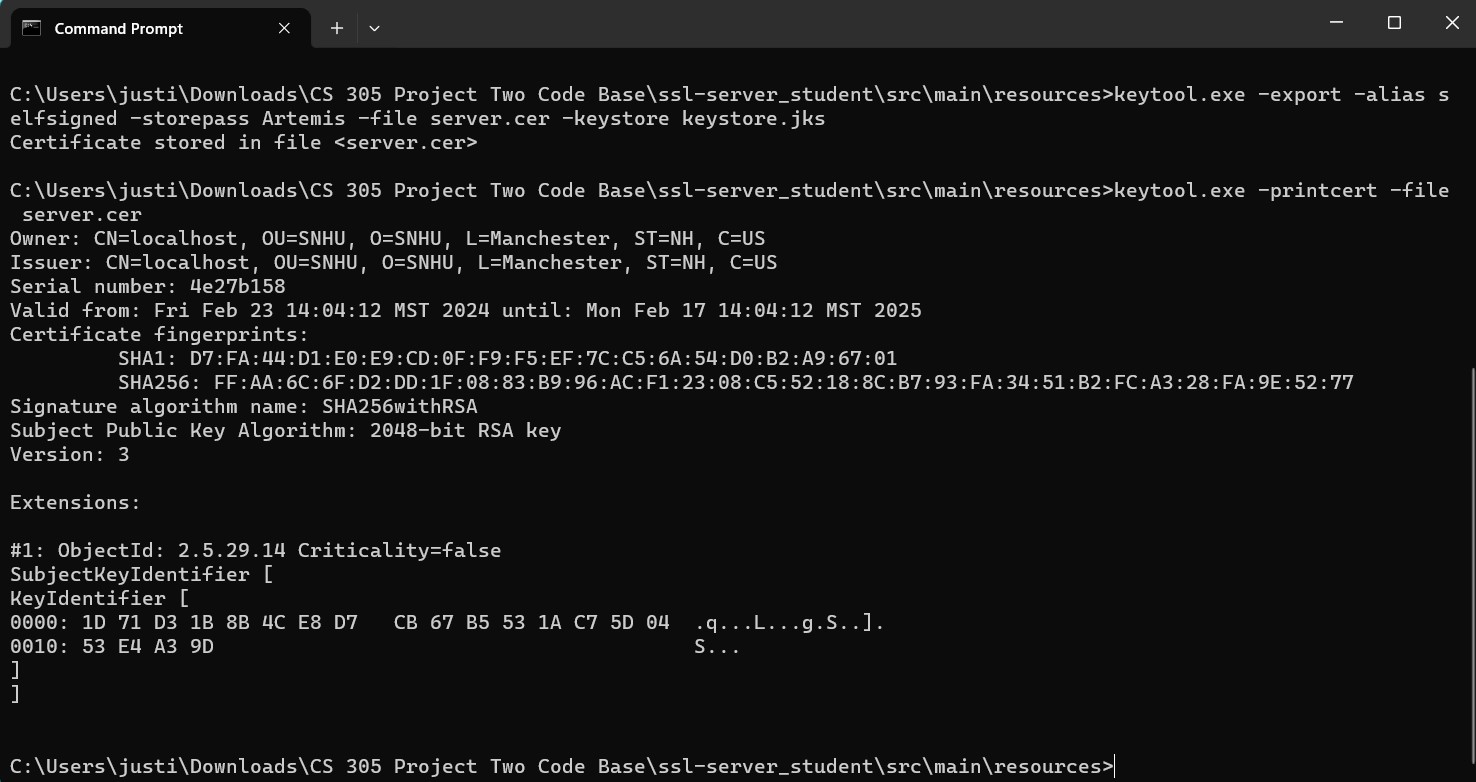
Justin Farquhar

## Algorithm Cipher

Artemis Financia is a consulting company developing individualized financial plans for customers, including savings, retirement, investments, and insurance. Because of this, I would recommend the use of SHA-256 encryption. It is really a cryptographic hash function, which is a one-way process, meaning that the original data cannot be retrieved from the hash value. As it is designed, every hash function is irreversible. Meaning a plaintext should not be produced if you have the digest beforehand and the digest will not provide the original value if you were to pass it through the hash function again. The final hash digest value will always be the 256 bits. The algorithm processes the data through 64 rounds of operations, making it extremely improbable to decrypt the message unless you were to know the series of operations performed. SHA-256 uses symmetric keys, using the initial algorithm to encrypt, then the same key reversed to decrypt. The algorithm computes the hash over a random number and then concatenates with a 64 bit counter. SHA-2 is still widely considered extremely secure, though the NIST has developed SHA -3 should it ever need to be replaced. Until there is a quantum computer created with enough computational power to crack hash functions, SHA will remain secure.

## Certificate Generation

Insert a screenshot below of the CER file.



## 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 screenshot of a computer

Description automatically generated

A screenshot of a computer code

Description automatically generated

## Secondary Testing

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

A screenshot of a computer

Description automatically generated

A screenshot of a check

Description automatically generated

## Functional Testing

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

A screenshot of a computer

Description automatically generated

## Summary

I Initially started by updating the versions of spring framework to 3.2.2, java to 17, and the maven dependency check to 9.0.9. With the older versions, the dependency check would show an excess of vulnerabilities that ultimately boil down to software being out of date, leaving problems that have been fixed in future versions out in the open. I then implemented cryptography methods to perform the encryption. Through this process, I ensured that my code was free of errors and did not open up any additional vulnerabilities.

## Industry Standard Best Practices

By updating the versions of the software we are using, I was directly mitigating against known security vulnerabilities. Known issues get patched through updates, making it extremely important to keep up with them. SHA-256 alone is one of the best practices for maintaining security as we are still far away from any serious threats, and if anything serious were to arise, the NIST has SHA-3 ready to deploy. Keeping up and implementing the industry standard best practices can greatly improve the efforts of cyber security professionals, giving them an upper hand at securing their companies software. While there are more people using these common practices, which could lead to increased bad actors attempting to crack algorithms, there are just as many people attempting to continually subvert these attempts, protecting you even more.