****

# 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** | **02/20/2024** | **Matthew Dziewiecki** |  |

## 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

Matthew Dziewiecki

## Algorithm Cipher

Recommend an appropriate encryption algorithm cipher to deploy, given the security vulnerabilities, and justify your reasoning. Review the scenario and the supporting materials to support your recommendation. In your practices for secure software report, be sure to address the following:

* Provide a brief, high-level overview of the encryption algorithm cipher.
* Discuss the hash functions and bit levels of the cipher.
* Explain the use of random numbers, symmetric versus non-symmetric keys, and so on.
* Describe the history and current state of encryption algorithms.

Artemis Financial is looking for an encryption algorithm that will be used for encrypting long-term archive files. We should assume that the most likely attack vector that Artemis would encounter is a bad actor, who is attempting to access their files. The best way to protect those files would be for them to be encrypted even when stolen, rendering them useless. These files will not be traveling so there will be no need for Asymmetric keys to be used on them. These files can also be archived for long term since these files will not be needing to encrypt quickly. I would recommend the SHA-256 cipher algorithm with 356-bit keys for Artemis Financials’ needs and encrypting their files. The SHA-256 cipher algorithm is one of the most secure default algorithms that is widely used and available in all the standard versions of Java. The encryption also provides the highest level of encryption which is 256-bit. This means that the more bits there are, the more possible key combinations that are which makes it harder for brute-force attacks and less likely to cause any collisions. The SHA-356 cipher algorithm uses Symmetrical encryption keys which will work with the company’s needs as they are the only ones that will be accessing the files. The algorithm also efficiently uses Java’s random number generation which ensures that each file is encrypted as securely as possible. Using the random number generation, this allows for the algorithm to securely create non-reversible checksum that is still able to verify the authenticity of the file. The hash functions is to verify files will be using the SHA-256 cipher algorithm to create a checksum signature of the provided files.

## Certificate Generation

Insert a screenshot below of the CER file.

A computer screen with white text

Description automatically generated

## Deploy Cipher

Submit a screenshot of the checksum verification in your practices for secure software report. The screenshot must show your name and a unique data string that has been created.

A screenshot of a computer

Description automatically generated

## Secure Communications

Create a screenshot of the web browser that shows a secure webpage and include it in your practices for secure software report.

HTTPS is working but that my Certification isn’t official because it’s self-signed.

A screenshot of a computer

Description automatically generated

A screenshot of a certificate

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 program

Description automatically generated

A screenshot of a computer

Description automatically generated

## Functional Testing

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

A screen shot of a computer program

Description automatically generated

## Summary

Discuss how the code has been refactored and how it complies with security testing protocols. In the summary of your practices for secure software report, be sure to address the following:

* Refer to the Vulnerability Assessment Process Flow Diagram. Highlight the areas of security that you addressed by refactoring the code.
* Discuss your process for adding layers of security to the software application.

While refactoring my code, I added the secure RestController into the application which will be used as the secure controller for the hast RESTful endpoint. To address the secure coding concern that’s in the Vulnerability Assessment Process Flow Diagram, the ServerController class takes care of that. The hashing cipher I have decided to use for the code is the SHA-256 hash algorithm as it is very secure and there is little chance of there being a collision.

## Industry Standard Best Practices

Explain how you applied industry standard best practices for secure coding to mitigate against known security vulnerabilities. Be sure to address the following:

* Explain how you used industry standard best practices to maintain the software application’s current security.
* Explain the value of applying industry standard best practices for secure coding to the company’s overall wellbeing.

The best practice to maintain the software application’s current security is to do at least once a month dependency checks on the application. This will check for any new potential vulnerabilities that were either fixed or discovered which is going to help significantly with the company’s overall well-being making sure that it is up to date with the latest updates to protect against potential attacks. Keeping up with the pom.xml plugins by ensuring that they are all up to date which will provide the highest level of security it can.