****

# 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** | **12/09/2022** | **AlisGurung** |  |

## 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

Alis Gurung

## Algorithm Cipher

Artemis Financial is requesting an encryption algorithm recommendation that will be used to encrypt long-term archive files. We should assume that the most likely attack vector for these files will be bad actors somehow acquiring access to these files, so they should be encrypted such that they would be useless even if they were stolen. The files will not be transported anywhere, so there is no need for Asymmetric\* keys to be used. Additionally, there is no need for these files to be encrypted quickly, as they will be archived long-term. Therefore, I recommend using the SHA-256 cipher algorithm with 256-bit keys to encrypt these files. SHA-256 encryption is the most secure default option available within all standard installations of Java, since it provides the highest level of bitwise encryption (256-bit refers to the number of bits in the length of the key. More bits mean more possible key combinations, thus making the key harder to brute-force and less likely to have collisions). SHA-256 also uses Symmetrical\* encryption keys. This will be fine, as Artemis Financial will be the only party accessing these encrypted files. The SHA-256 algorithm also makes efficient use of Java’s random number generation to ensure that each encrypted file is as secure as possible. Using random numbers allows for the cipher to securely create a non-reversible checksum that still verifies the authenticity of the file/message.

The hash function to verify files will use the SHA-256 cipher to create a checksum signature of the provided message.

## Certificate Generation

Insert a screenshot below of the CER file.

Text

Description automatically generated

## Deploy Cipher

Insert a screenshot below of the checksum verification.

Graphical user interface, text

Description automatically generated

## Secure Communications

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

Graphical user interface, text

Description automatically generated

## Secondary Testing

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

Graphical user interface, text, application

Description automatically generated

## Functional Testing

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

Graphical user interface, text, application

Description automatically generated

## Summary

Discuss how the code has been refactored and how it complies with security testing protocols. Be sure to address the following:

The purpose of each section is note in the functional testing section. Specifically the refactoring adding a checksum for input validation and a self signed certificate to allow for the use of HTTPS protocol. These two refactoring’s along with updating the spring-boot version ensure that the code is secure from major risk and error. To get to these conclusions my process for adding security layers starts by understanding what is the product and what is it required to do. I identify what the data is and the data that would be transferred for the service to be effective. I then ask myself if there are any regulatory issues that are needed to be considered for the code. After that I consider, if, for the areas of security identified, the data or dependency is needed for the code to run. Finally I start adding the security that will best meet the needs of the software.

1. Refer to the Vulnerability Assessment Process Flow Diagram and highlight the areas of security that you addressed by refactoring the code.
2. Discuss your process for adding layers of security to the software application and the value that security adds to the company’s overall wellbeing.
3. Point out best practices for maintaining the current security of the software application to your customer.

## Industry Standard Best Practices

In my code refactoring, I’ve added a secure RestController to the application to serve as the secure controller for the hash RESTful endpoint. This ServerController class addresses the secure coding concern in the Vulnerability Assessment Diagram and fulfills that concern. I’ve chosen to use SHA-256 as the hashing cipher for this function, and the code is very minimal so as to reduce the potential attack surface.

I’ve also updated the version of the Maven Dependency check version from 5.3.0 to 6.0.5, so that the static dependency checking is as accurate and up to date as possible.

To maintain the current security of the application, I’d recommend that the dependency checker is run at least once or twice per month in order to check for new vulnerabilities that have been discovered so that they may be fixed. Additionally, updating the plugins in the pom.xml configuration file is necessary every so often to ensure that the plugins remain up to date.