****

# 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** | **8/17/2023** | **Tatiana Case** |  |

## 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

Tatiana Case

## 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.

\*Symmetrical encryption keys means that the algorithm uses the same key to encode/decode the data. Asymmetrical keys use a combination of a public key to encode data and a private key to decode the data.

## Certificate Generation

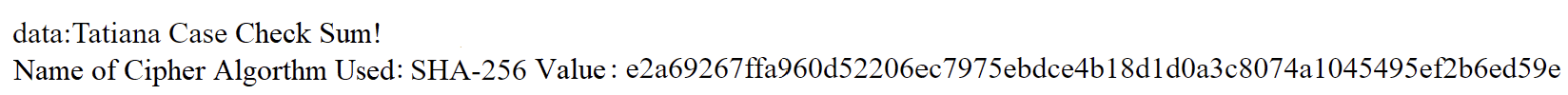
Insert a screenshot below of the CER file.

A screenshot of a computer

Description automatically generated

## Deploy Cipher

Insert a screenshot below of the checksum verification.



## Secure Communications

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

A screenshot of a computer

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 program

Description automatically generated

A white screen with text

Description automatically generated

## Functional Testing

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

A screenshot of a computer program

Description automatically generated

## Summary

During refactoring, I added a secure RestController to serve as the secure controller for the hash RESTful endpoint. The ServerController class addresses the secure coding concern in the Vulnerability Assessment Diagram and alleviates that concern. I chose to implement SHA-256 as the algorithm cipher for this function because it is simple yet secure. I was sure to include the updated version of the Maven Dependency check version from 5.3.0 to 8.4.0. This ensures that the static dependency checking is as accurate as possible.

To maintain the current security of the application, it is recommended that dependency checks be made at least once per month to test for new vulnerabilities. Also, updating the plugins in the pom.xml configuration file is necessary to maintain functionality.

## Industry Standard Best Practices

I implemented industry best practices when developing this application by utilizing proper naming conventions. Best practices include updating and patching any software that has updates available. Artemis Financial and their customers can rest assured secure coding was used at the highest standard. Vulnerabilities have been addressed and mitigated to ensure the environment is secure and safe to use. Taking the appropriate measures to protect the application ensures that Artemis Financial can safely grow their company without the constant fear of a breech. The value of approaching this project with security in mind increases the protection of the application from a malicious attacks by creating protocols to mitigate vulnerabilities.