****

# 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** | **[4/16/2023]** | **Ethan Fancher** |  |

## 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

Ethan Fancher

## Algorithm Cipher

Artemis Financial has requested a recommendation for an encryption algorithm to secure their long-term archive files. It's important to consider that the files may be accessed by unauthorized parties, so the encryption should render them useless even if stolen. As the files won't be transported, asymmetric keys are unnecessary, and there is no urgency to encrypt them quickly. I suggest using the SHA-256 cipher algorithm with 256-bit keys, which is the most secure default option available in Java. This algorithm uses symmetric encryption keys and makes efficient use of Java's random number generation to create a non-reversible checksum that verifies the authenticity of the file. The hash function will use the SHA-256 cipher to generate a checksum signature for the provided message. Since Artemis Financial will be the only party accessing these files, this algorithm will suffice

## 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, application, email

Description automatically generated

## Secure Communications

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

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

## Secondary Testing

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

Text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

## Functional Testing

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

Text

Description automatically generated

## Summary

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 used SHA-256 as the hashing cipher for this function, and the code is very minimal so as to reduce the potential attack surface.

## Industry Standard Best Practices

To maintain security and industry best practices, I would suggest once or twice monthly dependency checks of the application to keep the most up to date on potential vulnerabilities this will help to protect the company and their sensitive data. We’ve Implemented encryption to keep sensitive data protected