****

# 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** | **06-28-24** | **Joe Sullo** |  |

## 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

Joseph Sullo

## Algorithm Cipher

The most common attack that occurs in the financial sector is known as a bad actor attack, where someone attempts to access accounts or information without proper access. For this reason, encryption is the best solution for Artemis Financials’ security needs, and the SHA-256 encryption algorithm is the option in this realm. The “256” designation means that the algorithm generates a 256-bit hash key, twice that of the SHA-128 algorithm it was created to replace. While this makes it seem that 256 is twice as secure as 128, that is far from the case. Each additional bit doubles the available key space, meaning that 256 is roughly 2e128 times more secure than 128. By using a randomly generated string of lowercase letters and numbers, SHA creates a unique hash code for a piece of data which then requires a key to be read again. Systems with symmetric keys use the same for encryption and decryption while asymmetric keys are different for each step; asymmetric keys are more secure and are recommended for communications and transactions over the internet. Since SHA-256 can generate such a wide range of unique codes, it is very resilient to brute force attacks attempting to break its encryption. It also allows it to minimize collision issues, or two pieces of information having the same hash. This is extremely unlikely with SHA-256, to the point where it should never occur.

## Certificate Generation

Insert a screenshot below of the CER file.

A computer screen with text

Description automatically generated

A screenshot of a certificate

Description automatically generated

## 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.

HTTPS is active, throwing a not secure error because the certificate is self signed and not recognized by Microsoft Edge

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 screen shot of a computer program

Description automatically generated

A computer screen shot of text

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 computer screen shot of a building

Description automatically generated

## Summary

The most important step in adding security to the program was to enable the use of certificates, thereby enabling HTTPS, to ensure secure communication via the internet. This was done using a secure RestController to handle the program’s RESTful hash and to allow Spring to implement these RESTful web services. Proper certifications had to be created via the Java keytool application and stored properly to ensure that they could be accessed by the rest of the program. This is vital in the security of online communications because it ensures that the person, website, etc. (in this case website) that the client is communicating with is who they claim they are. As stated, SHA-256 was implemented into this RestController to provide hash codes and enable the security needed by Artemis Financial. This is important not only to safeguard the security of clients’ sensitive data but to also safeguard Artemis’s reputation as a safe financial institution. Once these two features were implemented, the final step was to test and manually review them, making sure that they function correctly.

## Industry Standard Best Practices

Using industry standard best practices is integral to proper security. Things such as syntax and naming conventions must be kept standard so that security functions will work and be accessible no matter where, when, or by whom the program was created. Exceptions and error messages must be properly implemented so that any problems that might arise can be quickly located and fixed, even by future developers of a project once it has been delivered to the customer. And the most important practice for security is to ensure that all of the plugins and dependencies that have been integrated into the program remain current and up to date. Hackers are constantly trying to devise new ways to break through security measures, and the finance industry is a prime target for such individuals. Developers work to patch any holes that might be discovered so that they cannot be exploited, but these updates will do no good if they are not checked for and applied regularly.