****

# 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** | **6/21/2024** | **Devin Criswell** | **First draft** |

## 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

Devin Criswell

## Algorithm Cipher

The algorithm cipher used for encryption was SHA-256. This is because of the information the client, Artemis Financial handles. SHA-256 is one of the safest and most difficult encryptions to break, which will provide our client with an added level of security. SHA-256 generates a hash value that is unique each time. The hash size for SHA-256 is 256 bits. It is the randomness of these 256 bits that makes it so difficult to hack. Without the private key the encrypted information is inaccessible, this is another benefit to using an asymmetric rather than a symmetric encryption.

## Certificate Generation

Insert a screenshot below of the CER file.

A screen shot of a computer

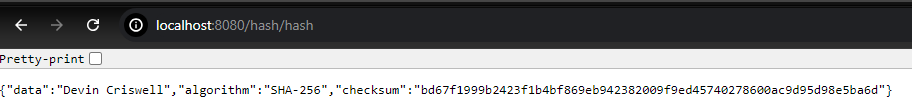
Description automatically generated

A black screen with white text

Description automatically generated

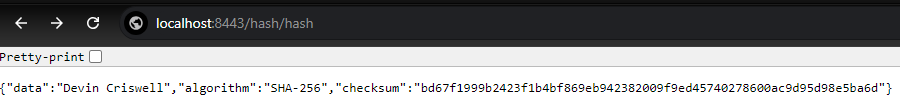
## Deploy Cipher

Insert a screenshot below of the checksum verification.



## Secure Communications

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



## Secondary Testing

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

A screenshot of a computer

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 screenshot of a computer

Description automatically generated

## Summary

To properly refactor the code a hash function was needed. SHA-256 was implemented for checksum verification to ensure the data’s integrity. Keytool was used to generate self-singed certificates, this will ensure the communication is encrypted. The certificate was exported to a CER file for use. The code was refactored in the application properties to use HTTPS. This will ensure that communication between the client and server is encrypted. I tested the software to ensure that the application was working as intended. A dependency-check was performed and reviewed to resolve any vulnerabilities.

## Industry Standard Best Practices

For industry best practices SHA-256 was implemented because it is secure and the most reliable hashing algorithm. HTTPS protocol was used to ensure all communications would be encrypted. A dependency-check was performed to find vulnerabilities and fix anything that was required. Testing was performed throughout development to ensure the application worked properly. The requirements were reviewed multiple times during development to ensure the application was meeting the set requirements.