****

# 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** | **12/14/24** | **Nicholas Gard** |  |

## 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

Nicholas Gard

## Algorithm Cipher

Artemis Financial is a banking company so the need for secure communication is paramount. Encryption can completely secure online transactions by scrambling the messages with a unique cipher. Asymmetric communication would be the best for a financial institution. Encryption can be accessed by anyone, but the decrypt key should remain private. The most secure and popular cipher right now is the SHA-256 algorithm. While it takes more time to de/encrypt it is an incredibly secure cipher that would be useful for a bank. A 256-bit key provides a near endless number of unique keys that are almost uncrackable. The SHA-256 makes use of Javas random number generator to ensure unique keys and a non-reversible checksum that verifies the validity of the file. The hash function will take an input data of variable length and convert it into a 32-byte has value. This represents a monumental advancement from the early days of cryptography where the US Government used 56-bit keys for military applications. Even in the modern day this doesn’t make algorithms undecipherable as technology like AI can process billions of unique keys given time and processing power.

## Certificate Generation

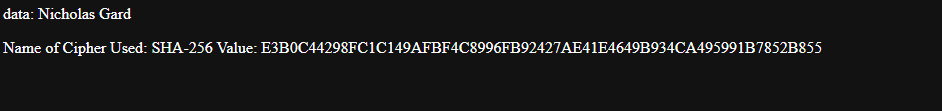
Insert a screenshot below of the CER file.

A screenshot of a computer screen

Description automatically generated

## Deploy Cipher

Insert a screenshot below of the checksum verification.



## Secure Communications

## Console line shows that the port was loaded with https

Webpage screenshot is the same as above.

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 screenshot of a computer program

Description automatically generated

## Functional Testing

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

A screen shot of a computer program

Description automatically generated

## Summary

To better secure my program I added a secured RestController as a secure controller for my programs RESTful stop. The ServerController class matches the problems given by the vulnerability assessment. In addition to that I used the SHA 256 cipher because it is one of the most secure and tested hash ciphers. It is a very complex and secure cipher that has a very small chance for collisions. Furthermore, to keep the program secure I would perform monthly dependency checks to search for emerging vulnerabilities and software updates. Securing the length of the pom.xml and keeping all plugins within it would also be beneficial to keep the program running at the highest level of security.

## Industry Standard Best Practices

I used many industry standards best practices to make sure this code is secure from infiltration. High level encryption with an industry standard cipher helps maintain the security of transmissions. A RESTful API keeps the software protected from malicious clients. Keeping the code simple and readable helps with security reviews and makes it easy to understand. These standards have high value to a project as there is a reason they are standard. They are tried and tested security measures that keep programs from being easily violated. Most security incidents happen from these standards being ignored, skipped or just done poorly.