****

# 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** | **August 15, 2024** | **Jordan Jenkins** |  |

## 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

Jordan Jenkins

## Algorithm Cipher

Artemis Financial is requesting Global Rain to add a file verification step to its web application. This is to ensure secure communications and provide a layer of security to data transferring. Artemis Financial is providing Global rain with their existing software application and requesting that Global Rain add on the proper security mechanisms to ensure the secure communications. I believe that the SHA-256 cipher algorithm (from the SHA-2 family) is the best algorithm cipher to use for such a project. The “256” in SHA-256 refers to the final hash1 digest value. Regardless of the size of the plaintext, the SHA-256 final hash digest value will always be 256 bits. The number of bits in SHA-256 makes it ideal from its predecessor, SHA-1. Where SHA-1 was starting to fail to keep up with brute force attacks, SHA-256 can help keep them at bay. Another benefit of the SHA-256 cipher is its use of a symmetrical key2. SHA-256 comes as a default option with Java and utilizes Java’s random number generation efficiently.

Something to note is that there are other ciphers in the SHA-2 family, such as SHA-512, which would provide more security. However, since there are more bits to digest, the cipher will be slower and take up more space. I think that the SHA-256 cipher will provide the security that Artemis Financial is looking for without sacrificing performance or space.

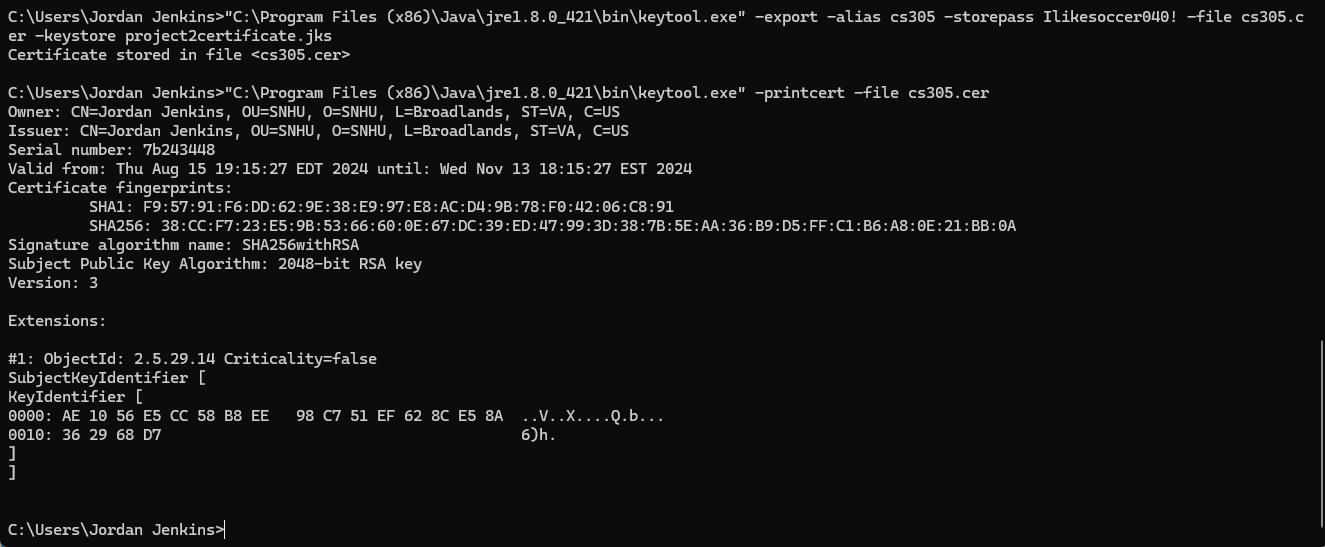
1A hash (or hashing) is the process of scrambling raw data to a point where the data cannot be put back into its original form. The data runs through a hash function that will perform mathematical operations to scramble the data and then output that information as a hash digest value.

Ex: “Hello World!” -> hash function -> J4%m@j04\*k

2A symmetrical key is an encryption key that is used for both encoding and decoding data. This is contrasts with an asymmetrical (or non-symmetrical) key that uses a different key to encode and decode data.

## Certificate Generation

Insert a screenshot below of the CER file.



## 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.

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

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 program

Description automatically generated

## Summary

For the refactored code, the RestController ensures that the code has a secure controller for the hash RESTful endpoint. The added ServerController class helps address the ‘Code Quality’ and ‘Code Error’ sections from the Vulnerability Assessment Process Flow Diagram (VAPFD). To satisfy the code quality, secure coding practices were followed via MessageDigest. Additionally, secure error handling was satisfied via NoSuchAlgorithmException. MessageDigest is the hash function that takes in the raw data and outputs the hash digest value. NoSuchAlgorithmException throws out errors when a requested element does not exist. There was troubleshooting of the pom.xml file and the application.properties that needed to be done to avoid failed builds and runtime errors. The printStackTrace method is a tool to help diagnose and handle the exceptions. The application.properties file needed to be updated with the appropriate information related to server.ssl.key-alias/store/store-password/store-type. Without this file updated, the refactored code would not be able to connect to the local host and exception errors and runtime errors would follow. Additionally, the pom.xml file needed to be updated for the maven dependency version. I updated the dependency-check-maven version to 10.0.3 to reflect the latest version on the OWASP Dependency-Check webpage. This helps provide an up-to-date dependency check report with accurate information to address vulnerabilities and suppress false positives.

Artemis Financial was requesting that Global Rain help ensure secure communications for their web application. SHA-256 was chosen because it utilizes hashing for data on web applications. SHA-256 takes the raw data and scrambles that data into a hash digest value. The reason for this bit size from the SHA-2 family and not something larger (like SHA-512), is that the larger the bit size, the more it weighs on performance and space. SHA-256 is more than enough security for Artemis Financial without sacrificing the performance of their web application or the storage.

To ensure continued security of the application, it is recommended that the maven dependency checker is run on the application at least once a month (if not more). This will help to keep the software up to date, assuming the appropriate dependencies are patched and added to the pom.xml file. Additionally, false positives can be filtered out by analyzing the dependency check report and adding vulnerabilities to a suppression.xml file. This will help save time when trying to address actual problematic vulnerabilities.

## Industry Standard Best Practices

To keep with the existing security in the application, I adjusted the version to 10.0.3 for the dependency-check-maven from OWASP. Additionally, I added import java.security.MessageDigest & import java.security.NoSuchAlgorithmException to import the appropriate packages to ensure that the data is hashed and the code can handle exceptions if elements are mising. I also tried to add in comments to show where the hash function and the exception handling is happening. By applying industry standard best practices for secure coding, the company can benefit from reputability. This would help retain clients and show that the company engages in industry standards and values quality and security.

*Noted files:*SslServerApplication.java – contains main application

application.properties – contains properties of the main application

pom.xml – contains configuration for project to build using Maven