****

# 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** | **6/9/2023** | **Jason Farrell** | **Initial Version** |

## 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

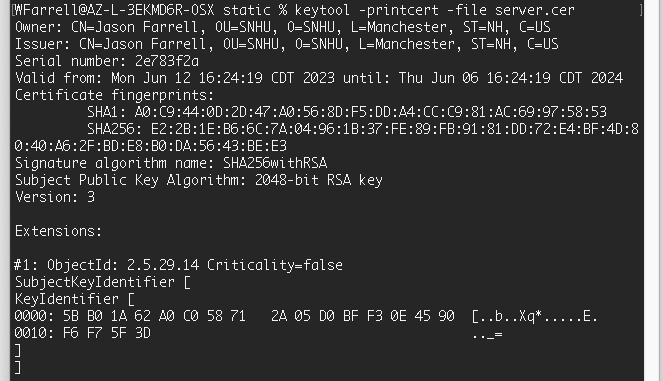
Jason Farrell

## Algorithm Cipher

I would recommend using the SHA-256 algorithm for hashing/encrypting all of Artemis Financial sensitive data due to their primary business focus dealing with global financial plans and customer’s accounts. SHA-256 as the name suggests (Secure Hash Algorithm) uses a 256-bit digest size (the size of the encrypted data after it has been hashed). The hashing logic transforms the data into a string of 256 bits using random numbers and padding bits which are appended to help obfuscate the message. SHA-256 is neither a symmetric nor asymmetric in that it doesn’t require public or private keys for encrypting and decrypting. In fact, the data hashed using SHA-256 cannot be decrypted as there are extremely low change of collisions. Once encrypted the data will stay secured. SHA-256 has proven itself since it was created in 2001 to be one of the best algorithm to use still today.

## Certificate Generation

Insert a screenshot below of the CER file.



## Deploy Cipher

Insert a screenshot below of the checksum verification.



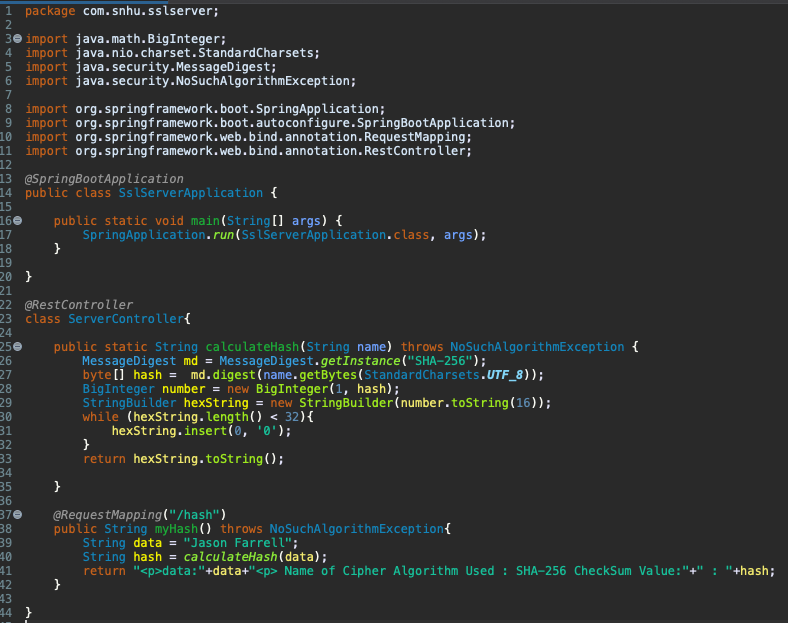
## Secure Communications

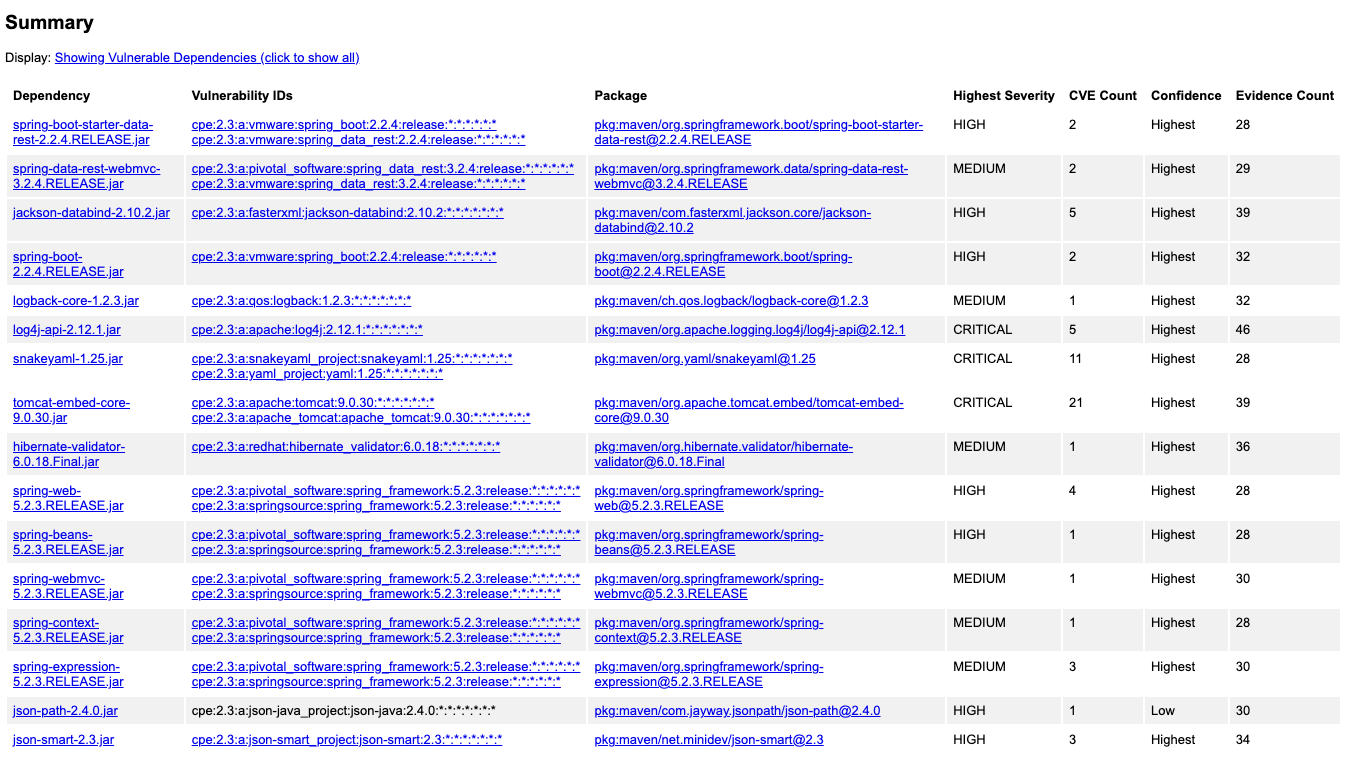
Insert a screenshot below of the web browser that shows a secure webpage.

The following screen was already secured however Chrome states my certificate is invalid due to no Certificate Authority. Here’s a screen shot of just using http  


## Secondary Testing

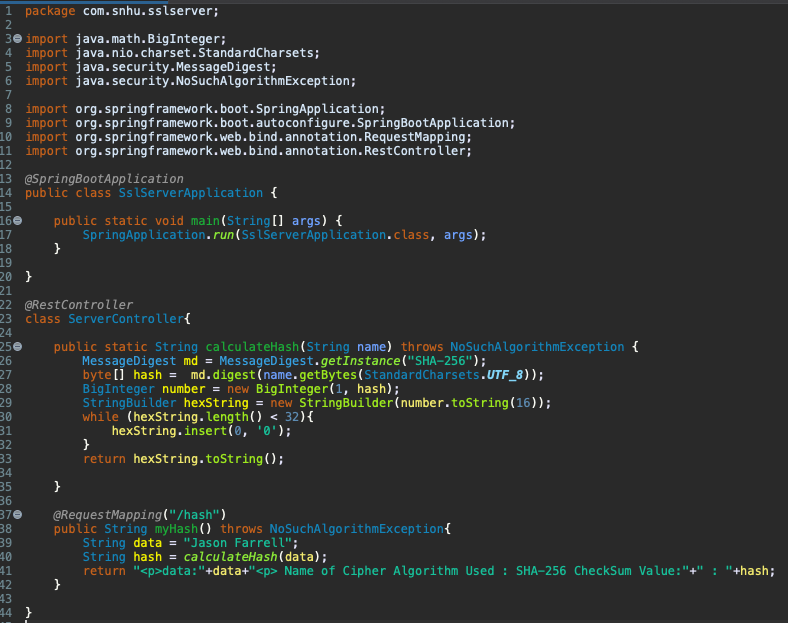
Insert screenshots below of the refactored code executed without errors and the dependency-check report.





## Functional Testing

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



## Summary

The Vulnerability Assessment Process Flow for a financial institution such as Artemis Financial dictates that we should focus on Input Validation, API’s, Code Quality and Encryption. The previous screen shot shows that we don’t have any input to validate yet using the API other than that it should be https SSL encrypted using a self-signed certificate which should later be changed to a certificate by a CA. Code quality can be seen by that there’s no warnings or errors being ignored in the code. Encryption is used by both https being enabled as well as a hashing algorithm which can be later used to hash files saved in remote and local file storage.

## Industry Standard Best Practices

I was able to use Oracle’s keytool provided with their JDK to create a keystore and to be created with an RSA SHA-256 self-signed certificate. I was also able to integrate a static testing tool by using the OWASP Dependency Check Maven plugin. Integrating industry standard practices is super important since they are proven methods and pattern for secure software engineering and at keeping platforms safe from bad actors. It’s important to keep up with industry standard practices due to new tools and techniques are being added every day to help keep systems and applications secure from vulnerabilities.