****

# 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** | **16 Oct 2023** | **Nur Faizah** |  |

## Client



## Developer

Nur Faizah

## Algorithm Cipher

Having been asked by Artemis Financial for a recommendation on an encryption technique designed for long-term archiving files. Unauthorised individuals accessing these files is the main threat.

Therefore, it is crucial to make sure that these files continue to be incomprehensible even if they are stolen. The requirement for asymmetric keys is not as prominent since these files likely won't be transferred and will only be kept for a long time.

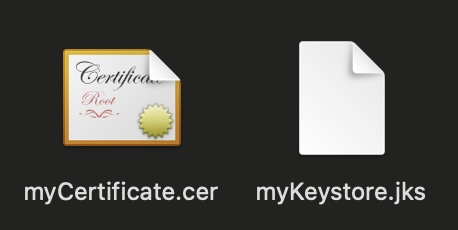
In light of the situation, I suggest using the SHA-256 cypher algorithm and 256-bit keys. The most secure encryption level is typically included in Java installations and is regarded as the highest level of security.

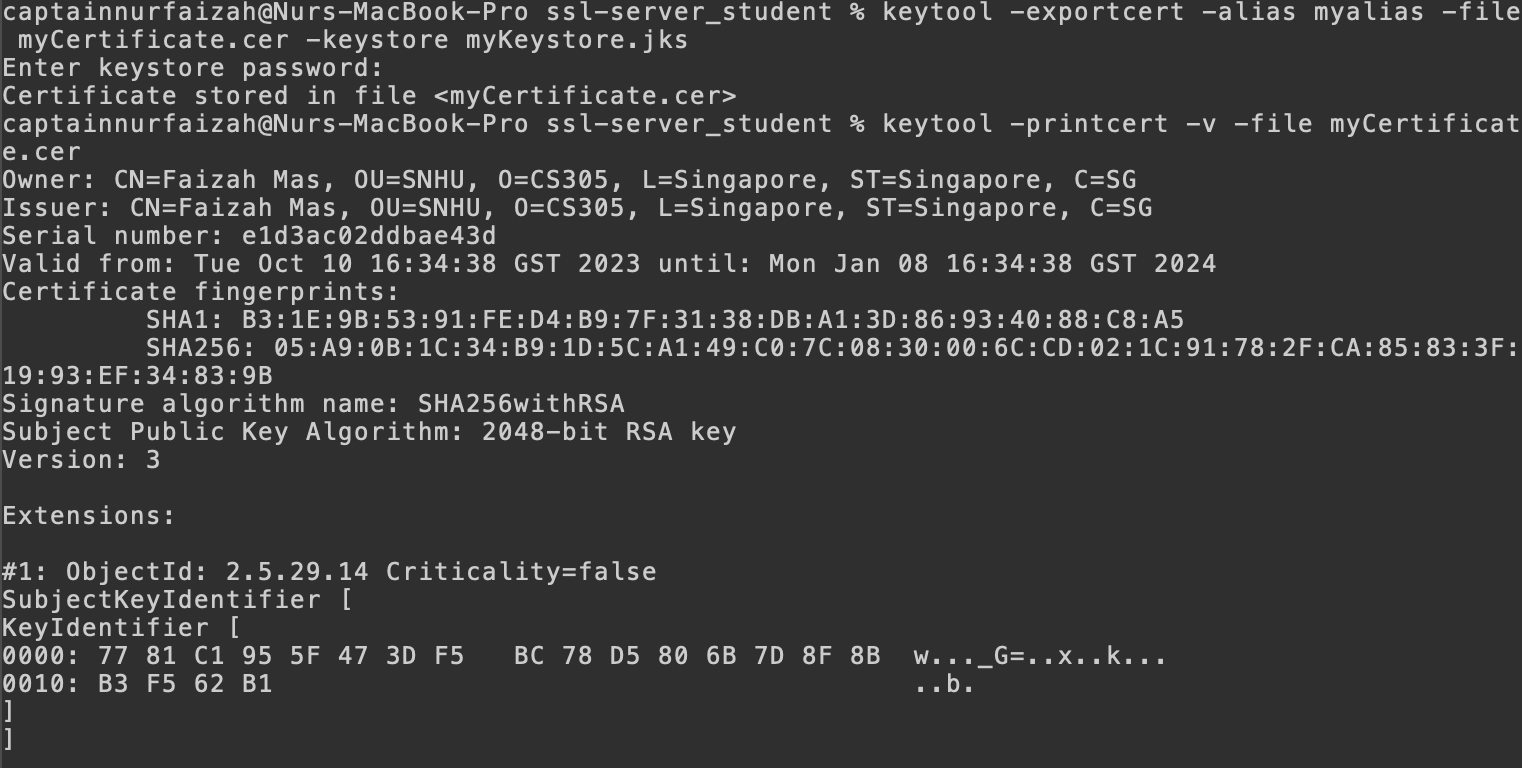
The 256-bit designation denotes the length of the key, which translates to a greater variety of possible key combinations, strengthening the resistance to brute-force attacks and decreasing the possibility of collisions.

It should be noted that SHA-256 uses symmetrical encryption, which is a wise decision given that only Artemis Financial plans to access these protected files.

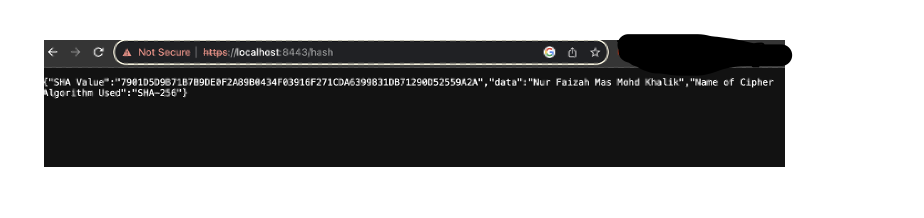
The algorithm also conveniently makes use of Java's random number capabilities to ensure that each encrypted file achieves the highest level of security. These random numbers give the cypher the ability to painstakingly create a non-reversible checksum while maintaining the integrity of the file or message.

## Certificate Generation.





## Deploy Cipher

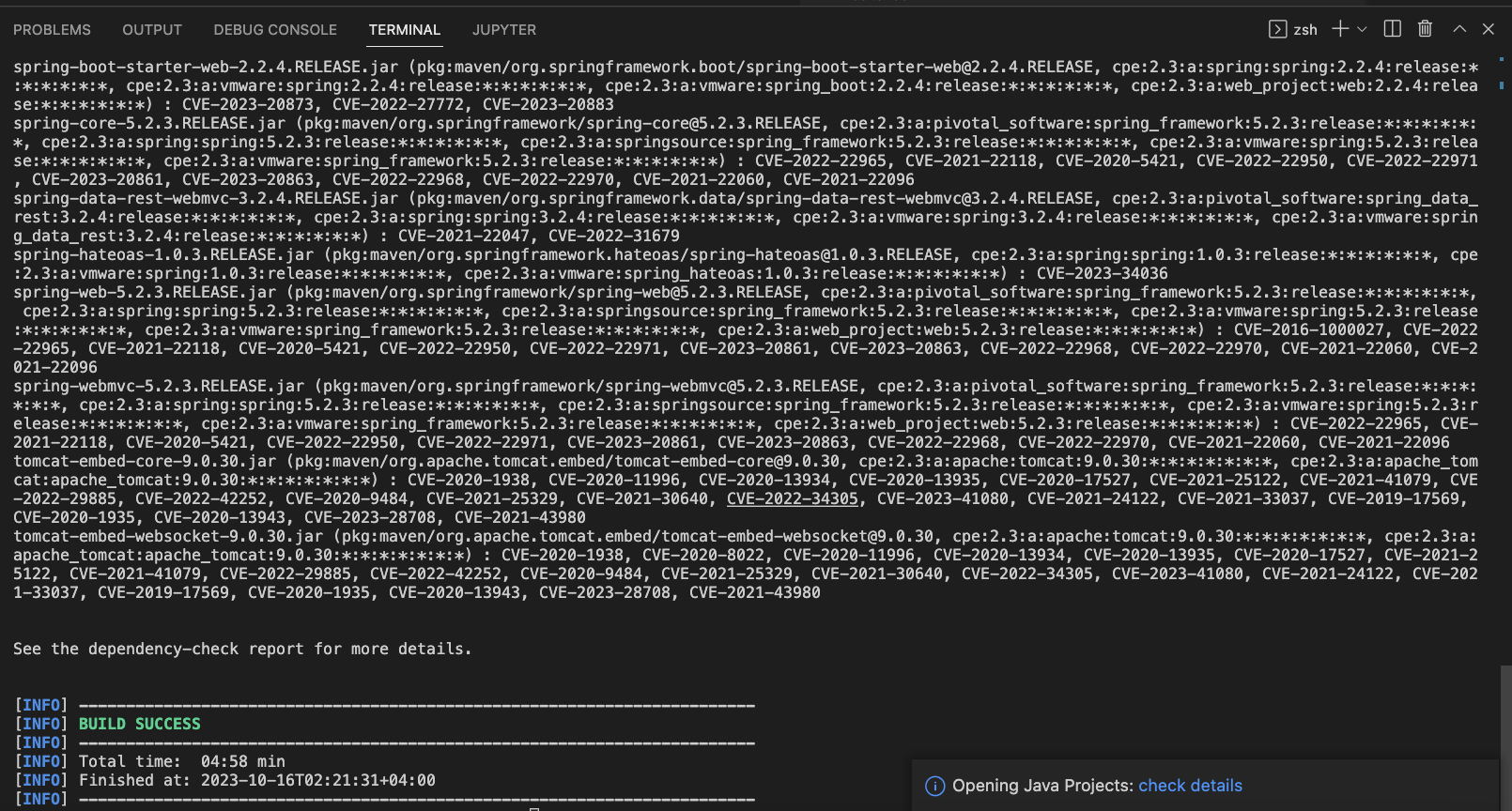


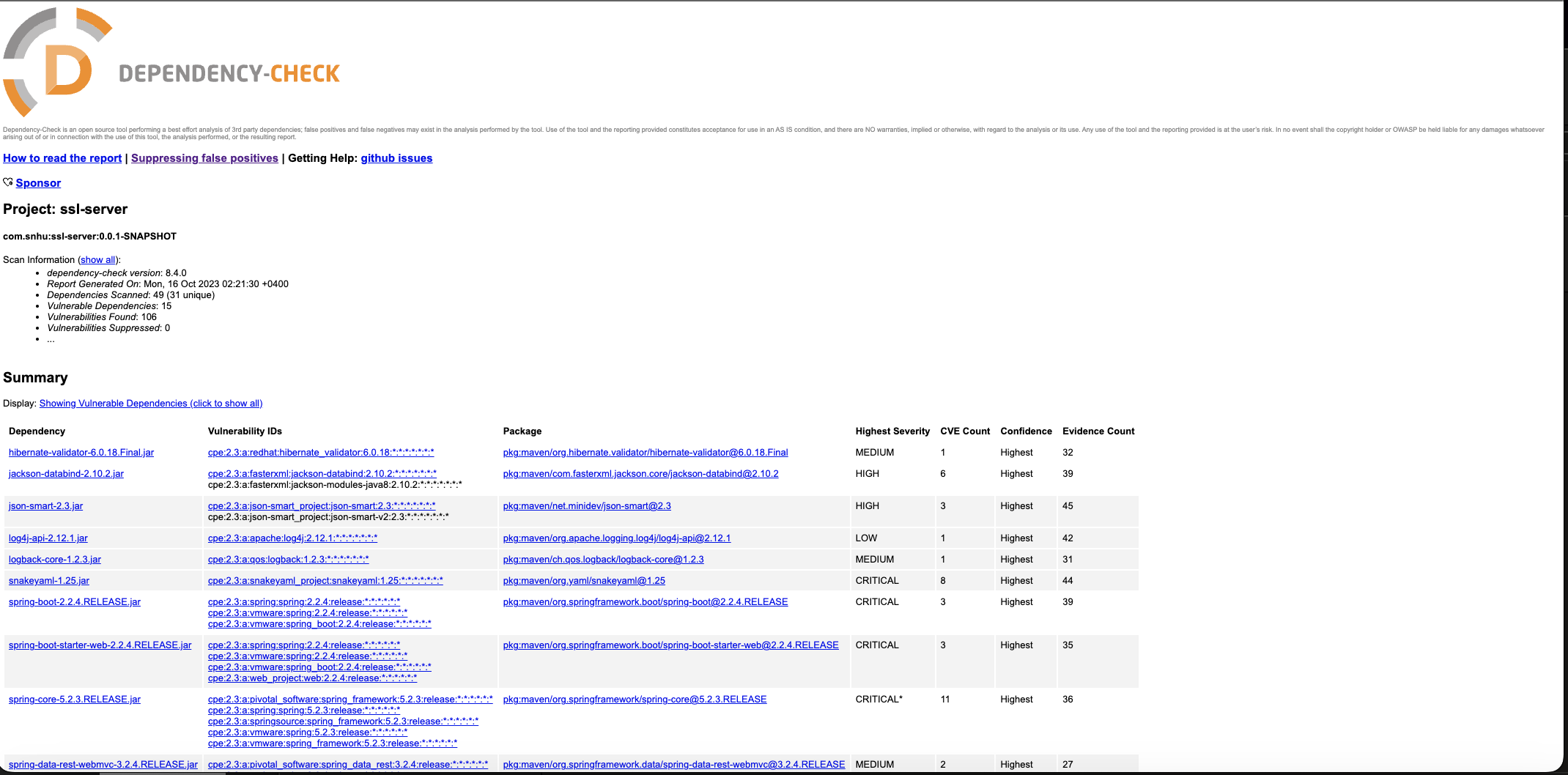
## Secure Communications

A black rectangular object with white text

Description automatically generated

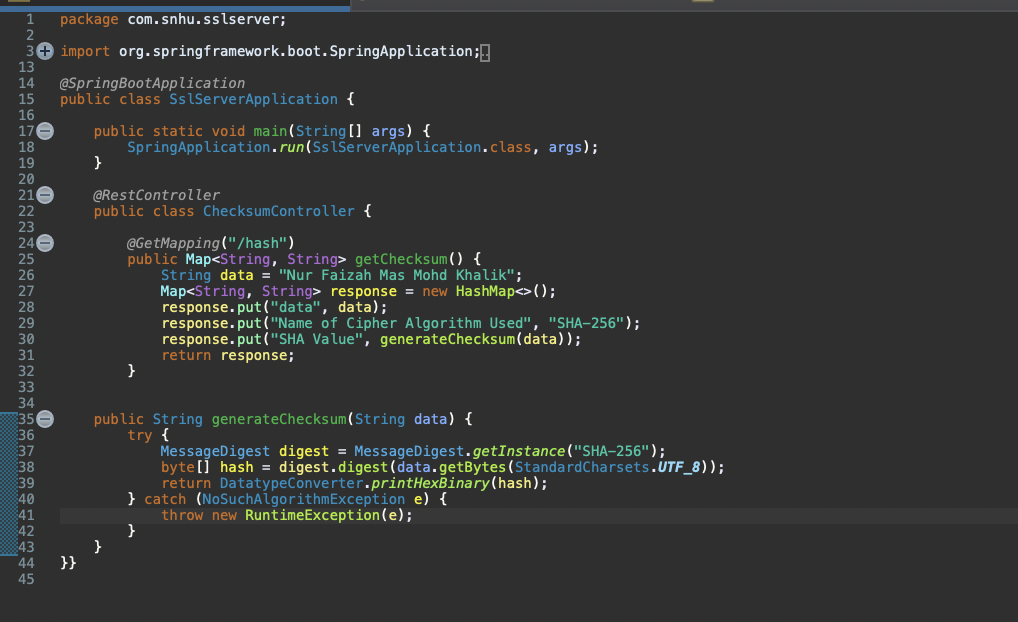
## Secondary Testing





## 

## Functional Testing



## Summary

SslServerApplication has gone through extensive refactoring to enhance its security mechanisms, guided by the provided Vulnerability Assessment Process Flow Diagram, ensuring a systematic and comprehensive approach to enhancing security. The adoption of the SHA-256 algorithm protects the application from vulnerabilities, particularly cryptographic ones. In addition, SHA-256 enhances exception-handling mechanisms, ensuring that anything out of the ordinary is swiftly taken care of.

## Industry Standard Best Practices

I closely followed industry best practices to maintain the security of the software. The application primarily employs SHA-256, which is a cryptographic hash function favored upon for its resistance to collision attacks. By doing this I ensured data integrity through the hashing function, thus enhancing the application’s security. In addition, the software uses `StandardCharsets.UTF\_8` for consistent character encoding, which reduces the vulnerabilities that stem from characters. These practices not only safeguards the application from vulnerabilities, but also ensures that the company’s reputation and trustworthiness is not tarnished by issues arising from potential security breaches.