

# 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](#_Toc102040759) 5

[3. Deploy Cipher](#_Toc102040760) 5

[4. Secure Communications](#_Toc102040761) 6

[5. Secondary Testing](#_Toc102040762) 6

[6. Functional Testing](#_Toc102040763) 7

[7. Summary](#_Toc102040764) 7

[8. Industry Standard Best Practices](#_Toc102040765) 7

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **22-Jun-24** | **Stephen Chryn** | **Initial Revision** |

## 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

Stephen Chryn

## Algorithm Cipher

There are many security protection best practices to consider when defending against various types of security attacks. The encryption strength is pretty important in protecting against brute-force attacks, and key management practices like secure key storage and rotation, are also up there in terms of importance. Integrity protection is good for ensuring that data is not only encrypted but also safe from tampering. The encryption solution Artemis should choose must also comply with relevant regulations, such as GDPR, PCI DSS, and others.

There are some risks with the recommendation above. The complexity might lead to errors that introduce vulnerabilities. Strong encryption algorithms can also create performance overhead, which might affecting system efficiency, or cause users to bypass necessary steps. Poor key management practices can lead to key leakage or loss; making the encryption ineffective or even cause data loss.

Compliance with government regulations is another big factor. GDPR requires the encryption of personal data, and PCI DSS mandates the encryption of cardholder data (Guest Author, 2023). The encryption algorithm will be used for long-term data storage, ensuring confidentiality and integrity over extended periods.

The best cipher recommendation for Artemis Financial is the Advanced Encryption Standard. AES is widely used and recognized as the industry standard for securing sensitive data. It offers strong protection against brute-force attacks, with AES-256 providing fantastic security. AES is efficient and, like all good tools, has hardware acceleration support on most modern processors. This means it is fast, strong, and reliable. It also meets the encryption requirements of various regulations, including FIPS 197 (FIPS 197, 2001).

## Justification of Encryption Algorithm Cipher

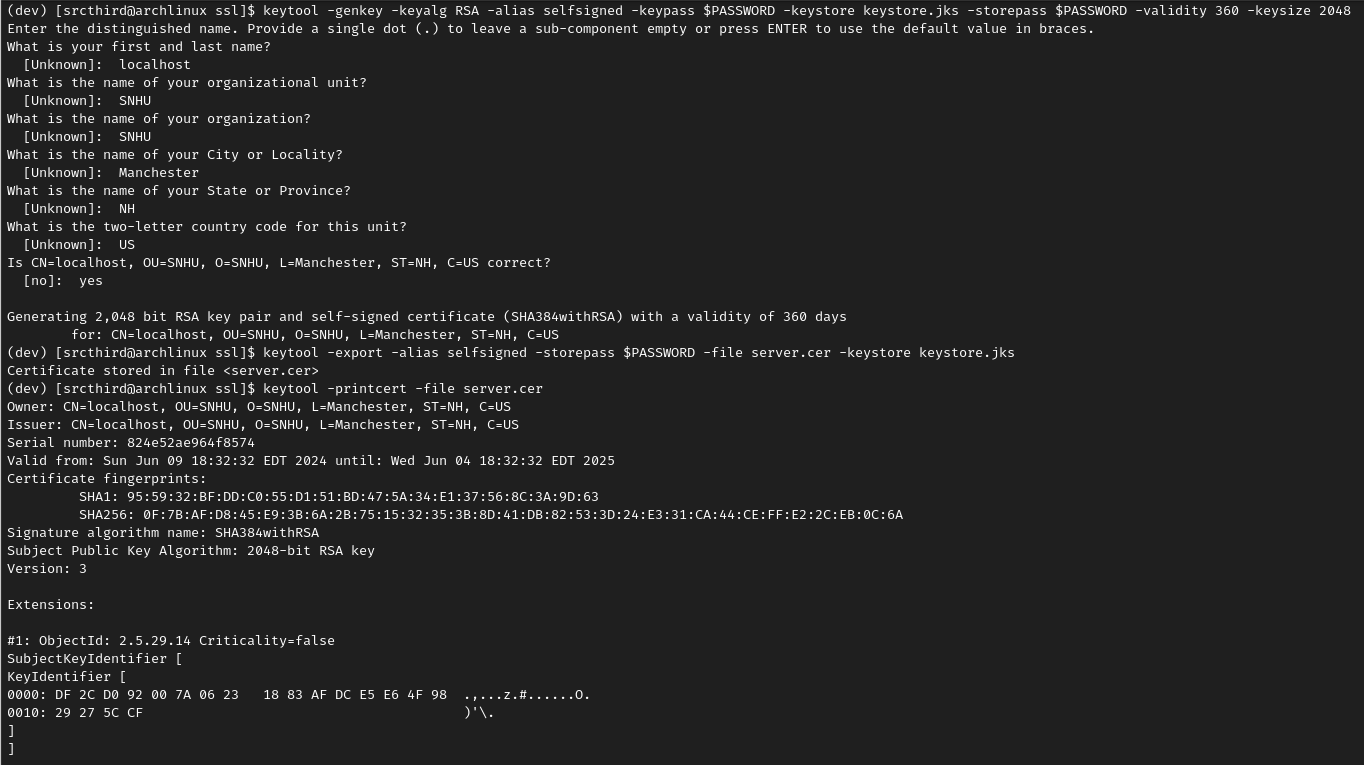
The purpose of the cipher's hash functions and bit levels is to enhance security. Hash functions are used to verify the integrity of data, ensuring it has not been tampered with, and produce unique outputs for different inputs. Higher bit levels, such as 256-bit keys, provide stronger security against brute-force attacks and ensure the encryption remains secure against future attacks (Team, 2024).

Random numbers, and symmetric versus non-symmetric keys play a huge role in encryption. Symmetric and asymmetric keys are also essential. Symmetric keys, like those used in AES, use the same key for both encryption and decryption, providing efficiency. Asymmetric keys, which use different keys for encryption and decryption, offer more security but are less efficient (FIPS 197, 2001).

The history and current state of encryption algorithms make cryptographic practices what they are today. Early encryption techniques, such as the Caesar cipher and Vigenère cipher, paved the way to the Data Encryption Standard in the late 1970s. DES was widely used until it was substituted with more secure options, due to its short key length of 56 bits. In 2000, the AES was adopted and is currently the most widely used encryption algorithm for securing data. Ongoing research in cryptography is focused on developing new algorithms resistant to quantum computing attacks (Sidhpurwala, 2024).

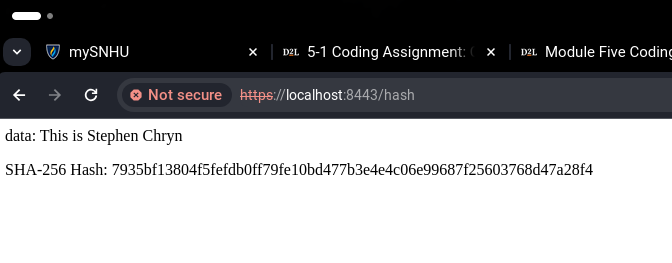
## Certificate Generation

Insert a screenshot below of the CER file.



## Deploy Cipher

Insert a screenshot below of the checksum verification.

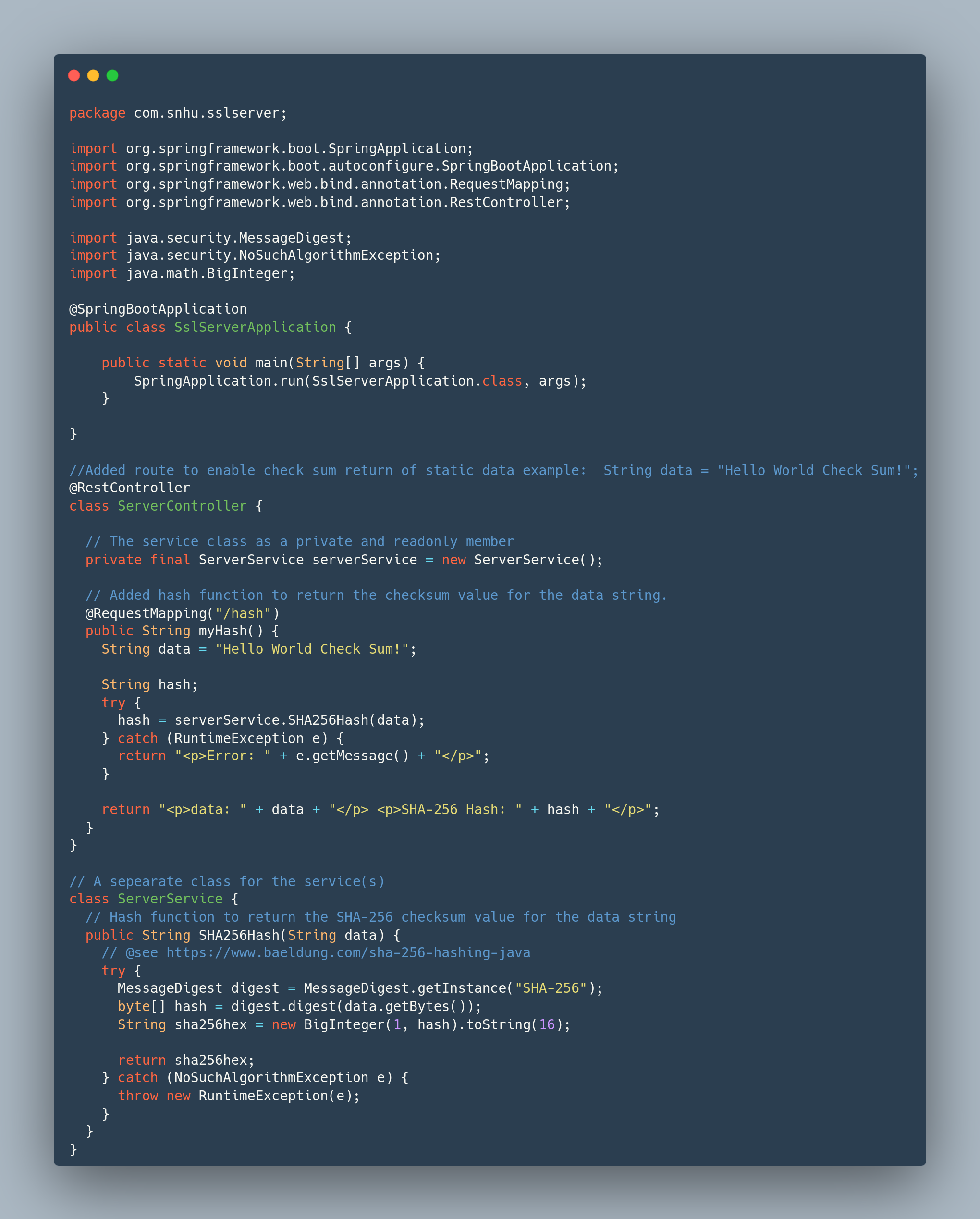
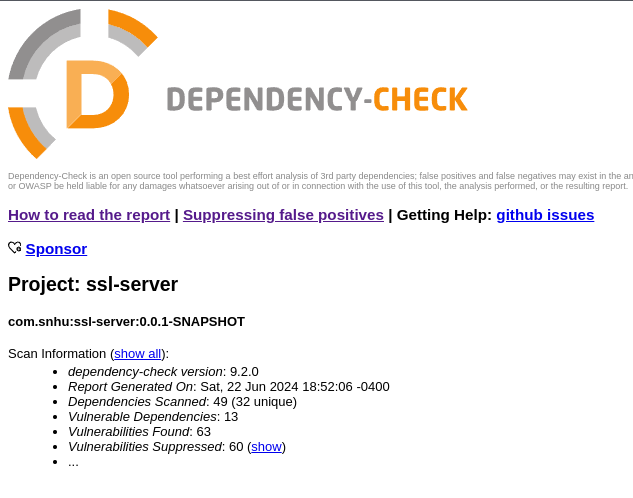


## Secure Communications

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

## 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

I refactored the code by introducing a new ServerController class annotated with @RestController to handle HTTP requests and a ServerService class to encapsulate the SHA-256 hashing logic. The ServerController includes a route, /hash, that processes a static data string, "Hello World Check Sum!", and returns its SHA-256 checksum. The hashing logic was moved to the ServerService class, ensuring separation of concerns and improving code readability and maintainability. This structure allows for easier future expansions and better testability by isolating the business logic from the controller.

I then configured SSL in the project by adding the key and certificate file to resources and updating the application.properties file to accept the SSL key and certificate under the alias "tomcat." This ensures that the web application can securely encrypt data transmitted from the API and decrypt data received from the client, utilizing the same certificate for both operations.

## Industry Standard Best Practices

In our Java Spring-Boot application, I adhered to industry-standard best practices by implementing a secure SHA-256 hashing function to ensure data integrity. By separating concerns through the use of a dedicated service class for hashing, and incorporating error handling, the application becomes strong, reliability and expandable. This secure coding practice not only enhances data security but also mitigates risks associated with data tampering. Adopting these industry-standard best practices in the development process strengthens the company's overall security, ensuring reliable and secure software that can enhance the company's reputation, reduces vulnerability to attacks, and creates trust among users.

Resources:

[*Java Security Standard Algorithm Names*](https://www.investopedia.com/news/cryptographic-hash-functions/). (2017). [https://docs.oracle.com/javase/9/docs/specs/security/standard-names.html#cipher-algorithm-names](https://docs.oracle.com/javase/9/docs/specs/security/standard-names.html" \l "cipher-algorithm-names)

Guest Author. (2023, September 12). *PCI DSS vs GDPR: Key Differences and Overlaps*. Cookie Law Info. <https://www.cookielawinfo.com/pci-dss-vs-gdpr-compliance/>

Federal Information Processing Standards Publication 197. (2001). In National Institute of Standards and Technology, Federal Information Processing Standards Publication. <https://csrc.nist.gov/files/pubs/fips/197/final/docs/fips-197.pdf>

Team, I. (2024, May 18). Cryptographic Hash Functions: definition and Examples. Investopedia. <https://www.investopedia.com/news/cryptographic-hash-functions/>

Sidhpurwala, H. (2024, April 30). A Brief History of Cryptography. Red Hat. <https://www.redhat.com/en/blog/brief-history-cryptography>