****

# 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** | **20 February 2024** | **Astrid French** | **Page 3 - 10** |

## 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

Astrid French

## Algorithm Cipher

After careful analysis, I have decided to use SHA256. Since this is for a checksum to ensure that the key the customer receives matches the code, a simple string checksum verification program is used. SHA-256 works by employing hash functions to secure data. The hash function acts as a data structure with a fixed-size hash value that uniquely represents the given data. This mechanism helps in data security and facilitates the detection of any potential data loss or alterations. Additionally, SHA-256 is known for its high collision resistance, meaning the probability of two different inputs producing the same hash output is extremely low. This property enhances the reliability of the checksum verification process, further ensuring the integrity and security of the transmitted data. A 256-bit version would be more suitable for enhanced security in data protection, especially since it is the most secure option and widely adopted by banking companies. In a cipher, hash functions are employed to secure data. The hash function acts as a data structure with a fixed-size hash value that uniquely represents the given data. This mechanism aids in data security and facilitates the detection of any potential data loss or alterations. On the other hand, bit levels refer to the length of the key used for encryption. Measured in bits, the key length directly impacts the strength of the encryption. Longer key lengths result in a more secure encryption, making it challenging for unauthorized access or hacker.

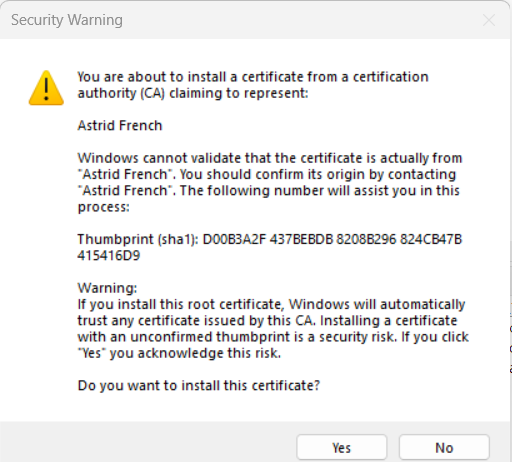
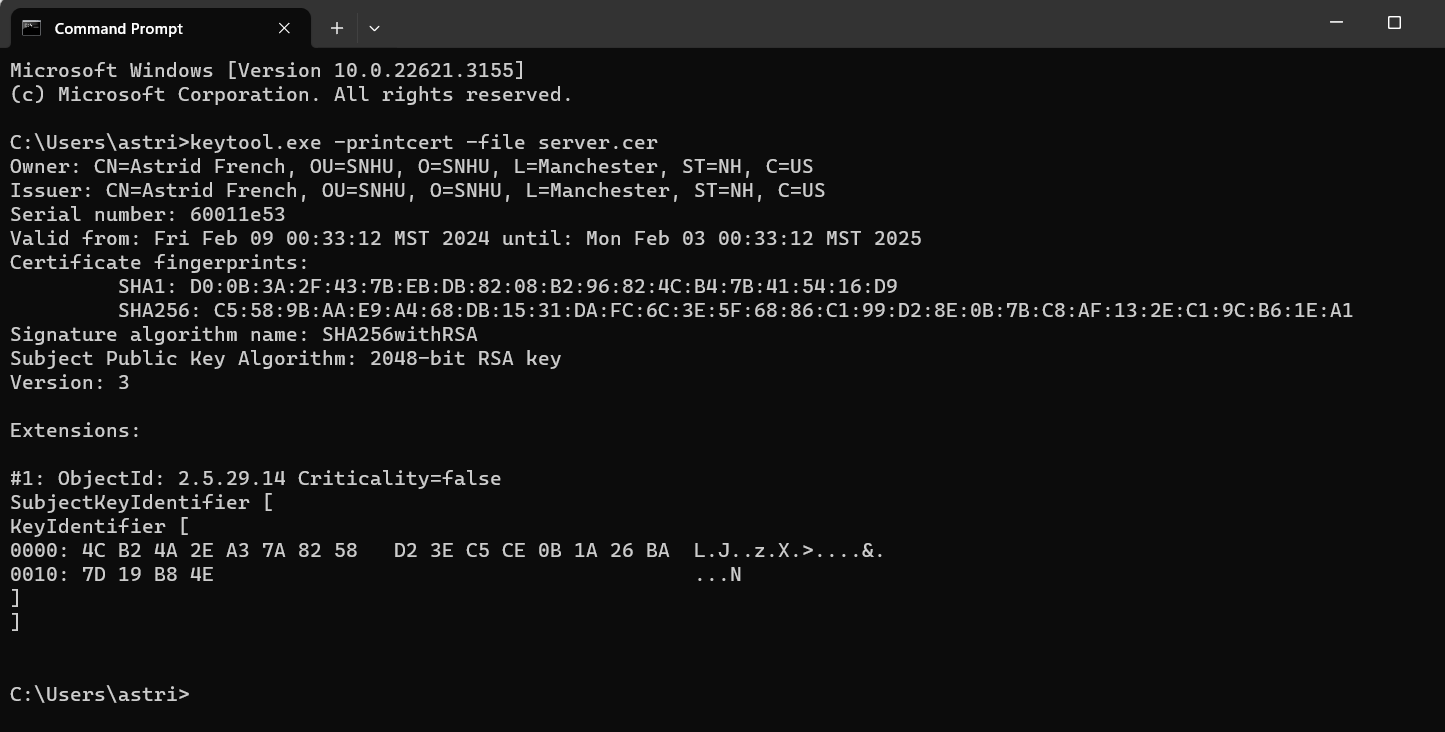
Random numbers are used in cryptography to make the system difficult to break into. While the symmetric keys and asymmetric keys used to secure the system during data transfer/communication. The difference “symmetric encryption uses the same key for both encryption and decryption, while asymmetric encryption uses a pair of keys: a public key for encryption and a private key for decryption” (Poggi, 2021).

Reference:

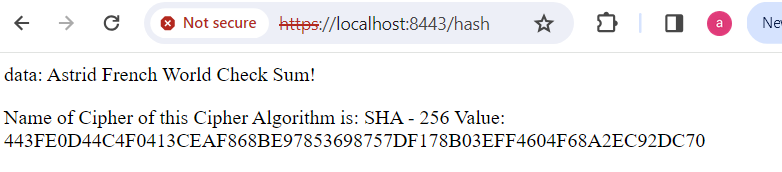
Poggi, N. (2021). *Types of Encryption: Symmetric or Asymmetric? RSA or AES?* Prey Project.

h<ttps://preyproject.com/blog/types-of-encryption-symmetric-or-asymmetric-rsa-or-a>es#:~:text=Asymmetric%20and%20symmetric%20encryption%20are,a%20private%20key%20for%20decryption.

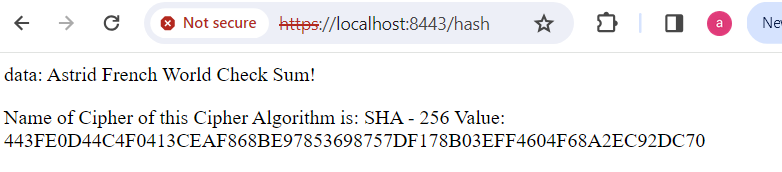
## Certificate Generation



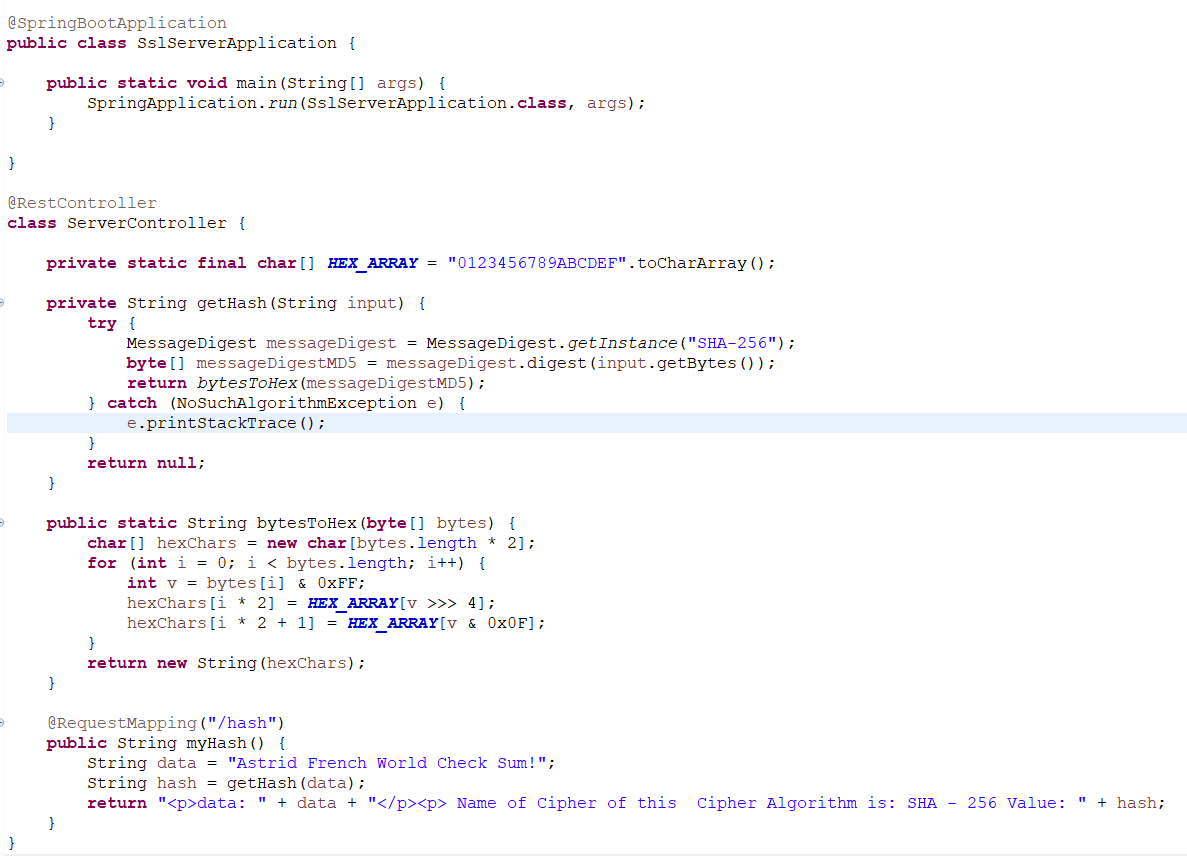
## Deploy Cipher



## Secure Communications



## Secondary Testing



## Functional Testing

## 

## 

**package** com.snhu.sslserver;

**import** java.security.MessageDigest;

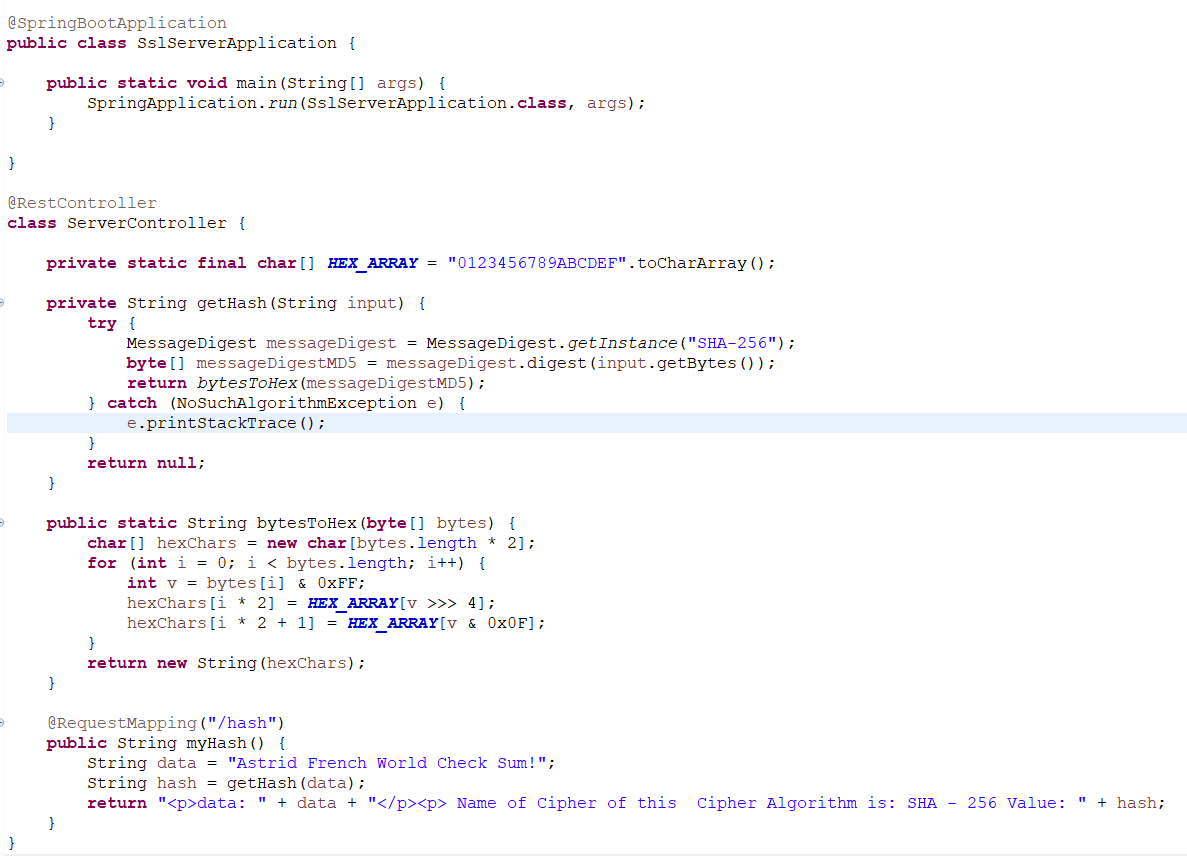
**import** java.security.NoSuchAlgorithmException;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

**import** org.springframework.web.bind.annotation.RequestMapping;

**import** org.springframework.web.bind.annotation.RestController;





## Summary

By refactoring my code, I have restricted access to only certain individuals who possess the password I specified. I updated and added several dependency data on my pom.xml:

* Updated my Maven dependencies to version 9.0.8 in my pom.xml file.
* Updated spring-boot-starter-parent to 3.0.13 version
* Updated Json-path to 2.9.0 version
* Updated logback-core to 1.4.14 version
* Updated Jackson-databind to 2.16.0 version
* Updated spring-core to 6.0.17 version
* Updated spring-web 6.0.17 version
* Added suppression.xml file to my pom.xml

Adjusted my application properties to align with the certificate generation process. Utilizing SHA-256 as my hashing cipher enhances security measures to prevent potential attacks. It's imperative to continuously monitor and update Maven dependencies and plugins to ensure strong protection against emerging threats. And my dependency-check end up with 0 or no vulnerability.



## Industry Standard Best Practices

Artemis Financial is a financial company that handles people's savings, loans, retirement, and other sensitive data. We prioritize the best protection for our company to gain trust from customers. Here are some industry standard best practices we follow:

1. Ensuring clean and error-free code inputs.
2. Input validation, to make sure it’s the right data that goes into the database
3. Implementing encryption measures.
4. Emphasizing authentication to verify the identity of users accessing data.
5. Regularly updating to the newest or most secure software versions.
6. Adhering to cryptographic practices for data security.

By implementing these practices, we aim to safeguard sensitive information, maintain customer trust, and uphold the integrity of our financial services.