****

# 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** | **12/09/2023** | **Israel Stillman** |  |

## 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

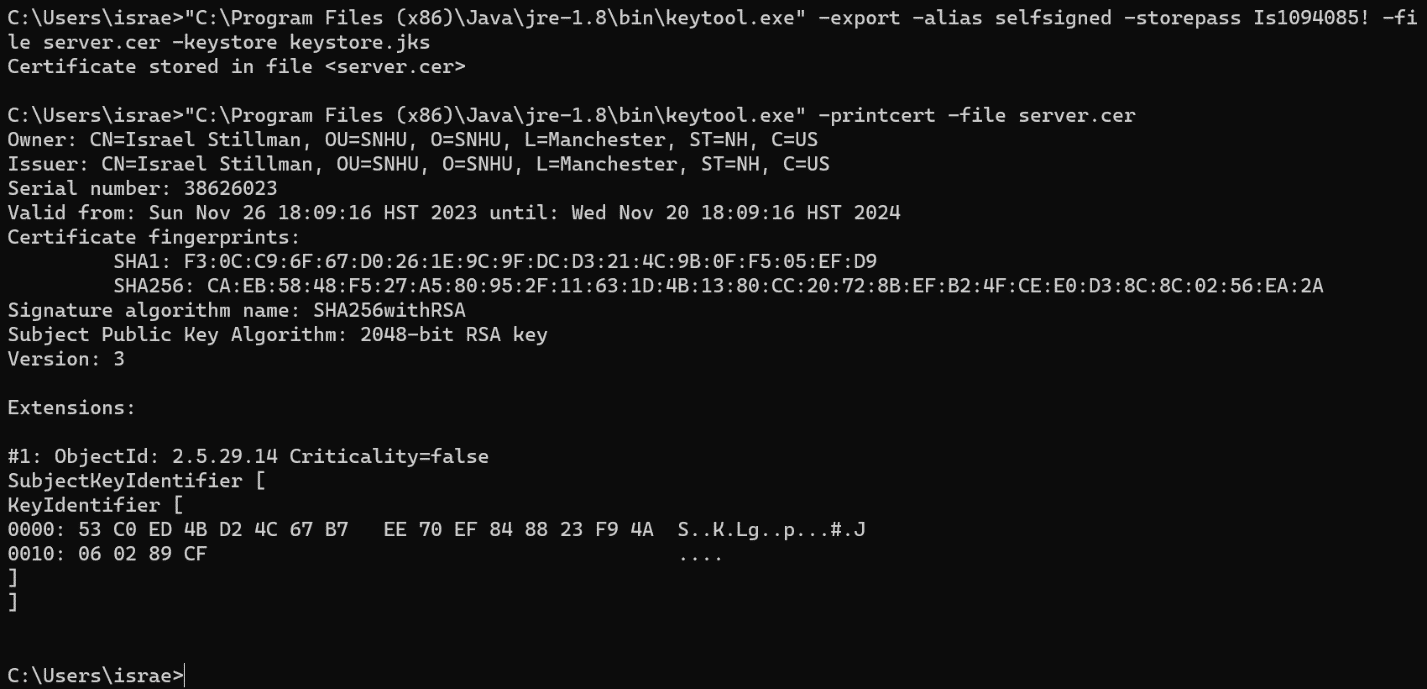
Israel Stillman

## Algorithm Cipher

I recommend employing the SHA-256 (Secure Hash Algorithm 256-bit) encryption algorithm cipher. SHA-256 is part of the Java Security Standard Algorithm Names provided by Oracle. It is well-regarded for its resistance to collision attacks, producing a 256-bit hash value that significantly enhances security layers for data integrity.

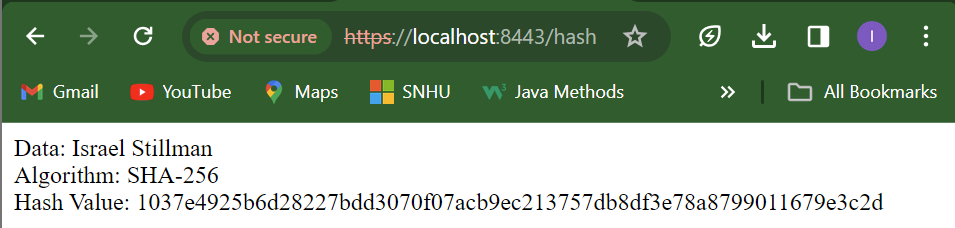
## Certificate Generation

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.

A screenshot of a computer

Description automatically generated

I wasn’t able to successfully verify a secure connection.

## Secondary Testing

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

A screen shot of a computer program

Description automatically generated



## Functional Testing

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

A screen shot of a computer program

Description automatically generated

## Summary

The addition of a secure RestController to handle sensitive endpoints, such as the hash RESTful endpoint, adresses concerns outlined in the vulnerability assessment. The implementation of the ServerController class mitigates potential security risks associated with these endpoints. By incorporating robust security measures, including the use of SHA-256 as the hashing cipher and maintaining minimalistic code structures, the attack surface can be reduced. The process of fortifying the software application involved systematically adding layers of security. This began with the utilization of the MessageDigest class from the java.security.MessageDigest library, enabling the creation of a secure hash value.

Each step of the refactoring process was aligned with security best practices, serving as an overarching strategy to reinforce the software's resilience against potential threats. This approach enabled the integration of multiple security layers while preserving system functionality and performance. Ultimately, the culmination of these efforts was encapsulated in a comprehensive and robust security framework that adheres to stringent testing protocols, significantly enhancing the application's resistance to potential security breaches.

## Industry Standard Best Practices

To maintain the application's current security, I followed established industry guidelines and frameworks such as OWASP (Open Web Application Security Project), the addition of a secure RestController aligned with the principle of least privilege, ensuring sensitive endpoints like the hash RESTful endpoint were accessible only through authenticated and authorized channels. Implementing the ServerController class enforced a robust encryption scheme, utilizing the SHA-256 hashing cipher to bolster data security.

By prioritizing secure coding practices, the company establishes a robust foundation, fostering trust among users and stakeholders. Furthermore, adherence to these practices cultivates a culture of security awareness and preparedness within the organization, fostering resilience in the face of evolving cyber threats.

The value of these practices extends beyond immediate security concerns, impacting the company's market competitiveness, regulatory compliance, and customer trust. It underscores a commitment to safeguarding sensitive information, maintaining data integrity, and ensuring continued business operations amidst an increasingly complex threat landscape. Ultimately, applying industry-standard best practices in secure coding bolsters the company's overall resilience, elevates its reputation, and instills confidence among stakeholders in the digital ecosystem.