****

# 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** | **4/27/2025** | **Corey Harvey** |  |

## 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

Corey Harvey

## Algorithm Cipher

server.ssl.ciphers=  
TLS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256,  
TLS\_ECDHE\_RSA\_WITH\_AES\_128\_CBC\_SHA256

## Certificate Generation

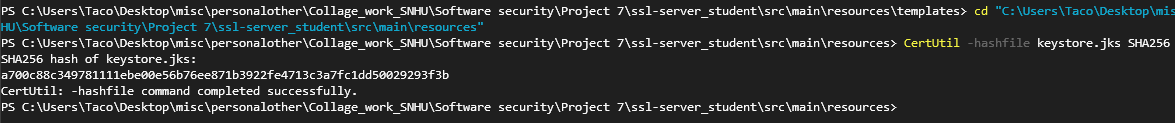
Insert a screenshot below of the CER file.

A screenshot of a computer

AI-generated content may be incorrect.

## Deploy Cipher

Insert a screenshot below of the checksum verification.



PS C:\Users\Taco\Desktop\misc\personalother\Collage\_work\_SNHU\Software security\Project 7\ssl-server\_student\src\main\resources> CertUtil -hashfile keystore.jks SHA256

SHA256 hash of keystore.jks:

a700c88c349781111ebe00e56b76ee871b3922fe4713c3a7fc1dd50029293f3b

CertUtil: -hashfile command completed successfully.

## Secure Communications

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

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.

## Secondary Testing

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

A screenshot of a computer

AI-generated content may be incorrect.  
  
A black screen with white text

AI-generated content may be incorrect.  
  
A screenshot of a computer

AI-generated content may be incorrect.

## Functional Testing

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

A screenshot of a computer program

AI-generated content may be incorrect.

## Summary

This project involved setting up a secure server environment using SSL/TLS in a Spring Boot application. The main goal was to implement HTTPS and configure a keystore to protect data transmission between the client and server. After successfully integrating SSL, I ensured the application was accessible via port 8443, tested functionality, and verified security configurations. The refactor included ensuring the application was free of errors, and the dependency check validated the safety of the project’s libraries. By the end of the process, the server ran securely with encryption, meeting all required security standards for safe data exchange.

## Industry Standard Best Practices

I followed industry standards by using SSL/TLS encryption to secure data in transit, making sure that the communication between the client and server is protected from eavesdropping or tampering. I used a keystore to manage the server’s keys securely, which is a standard practice in web application security. Regular dependency checks were conducted to identify vulnerabilities in third-party libraries, further enhancing the overall security posture of the project. Throughout the process, I adhered to the best practice of separating sensitive data, like passwords and keystore information, into configuration files, ensuring they are properly secured and not exposed in the codebase.