****

# 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** | **12/07/2023** | **Corvinna Curtis** |  |

## 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

Corvinna Curtis

## Algorithm Cipher

In this secure software project, we have chosen to employ the Advanced Encryption Standard (AES) as our algorithm cipher. AES is renowned for its robust security features, making it an ideal choice for data transmission. AES operates on symmetric key cryptography, where both the sender and receiver utilize the same key for encryption and decryption. We have selected AES-256, which employs a 256-bit key size, ensuring a high level of security. The longer key size enhances security as it exponentially increases the number of possible keys, making brute-force attacks virtually impossible. AES-256 has a proven track record in securing data and is widely adopted in various security-sensitive applications.

## Certificate Generation

The certificate generation process is a critical step in establishing trust and authenticity within our system. We used the Java KeyStore (JKS) tool, which is a component of the Java Development Kit (JDK), to generate self-signed certificates. These certificates play a pivotal role in authenticating parties involved in secure communication. During the certificate generation, we created a Certificate Signing Request (CSR), including essential details such as the Common Name (CN), Organization, and Country. This information helps in identifying the certificate holder and their affiliation. The CSR was then signed by a trusted Certificate Authority (CA) to produce the final CER (Certificate) file. The involvement of a reputable CA enhances the trustworthiness of the certificates and ensures secure communication within our system.

A screenshot of a computer

Description automatically generated

A screenshot of a certificate

Description automatically generated

## Deploy Cipher

After deploying the cipher, we conducted a checksum verification process to ensure the integrity of the deployed cipher. This crucial step guarantees that the cipher is successfully deployed without any corruption. We employed cryptographic hash functions, such as SHA-256, to generate checksums from the cipher file. These checksums serve as unique fingerprints of the file and their verification ensures that the cipher remains unaltered during deployment. This added layer of security safeguards against potential tampering or unauthorized changes to the cipher, providing confidence in its integrity.A screenshot of a computer program

Description automatically generated

## Secure Communications

To ensure secure communications, it is imperative to configure the system to establish a secure HTTPS connection in the web browser. Secure communication is achieved through the use of Transport Layer Security (TLS) protocols, which encrypt data during transmission, safeguarding it from eavesdropping or tampering. We meticulously configured the server and web application to support HTTPS. This process involved obtaining and configuring SSL/TLS certificates, updating the application's properties file, and configuring the server to handle secure connections. While we encountered challenges during this process, including certificate installation and server configuration, these challenges were successfully overcome through careful implementation and adherence to industry-standard best practices.

A screen shot of a computer

Description automatically generated

## Secondary Testing

In this section, we conducted comprehensive secondary testing to validate the refactored code and generate a dependency-check report. The refactored code was executed without any errors, signifying the success of the code refactoring process. Functionalities and components added as part of the refactoring were thoroughly tested to ensure they perform as intended. Additionally, we employed the OWASP Dependency-Check Maven tool to identify and report any known vulnerabilities in third-party libraries and components used in the project. This proactive approach ensures that the application is not susceptible to common security vulnerabilities, especially those related to open-source libraries with known exploits.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

## Functional Testing

Functional testing is a critical phase in the software development process, ensuring that the application functions correctly and as expected. We conducted rigorous functional testing to verify that the refactored code performs its intended tasks accurately. This testing involved various scenarios, including edge cases and user interactions, to ensure that the application is robust and free from functional defects. Functional testing is crucial for delivering a reliable and secure software solution to end-users.

A screenshot of a computer

Description automatically generated

## Summary

This secure software development project followed a rigorous process to ensure the confidentiality and integrity of data during transmission. We implemented industry-standard security measures, including the use of the AES-256 encryption algorithm, meticulous certificate generation, checksum verification, and the establishment of secure web communication through HTTPS and TLS protocols. Challenges were encountered, including server configuration and certificate installation, but they were successfully overcome through careful implementation and adherence to industry-standard best practices. The result is a secure and reliable software solution that meets the highest standards of data protection and security.

## Industry Standard Best Practices

Throughout the project, we adhered to industry-standard best practices for secure software development, which included:

* Implementing encryption using AES-256 for data protection, leveraging its robustness and enhanced key size.
* Generating certificates with detailed information, ensuring authenticity, and relying on trusted Certificate Authorities for enhanced security.
* Conducting checksum verification to guarantee the integrity of the deployed cipher, mitigating tampering risks.
* Utilizing HTTPS and TLS protocols to establish secure web communication, safeguarding data during transmission.
* Performing comprehensive functional testing, including edge cases and user interactions, to deliver a reliable and secure software solution.
* Employing the OWASP Dependency-Check Maven tool to identify and address vulnerabilities in third-party components, ensuring proactive security measures.

By following these industry-standard best practices, we have developed a secure software solution that not only meets the highest standards of data protection and security but also contributes to the overall wellbeing of the company by safeguarding sensitive information and ensuring the trustworthiness of our software.