****

# 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** | **13/8/23** | **Austin Blanton** |  |

## 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

Austin Blanton

## Algorithm Cipher

**Recommendation**: AES (Advanced Encryption Standard)

* **High-level Overview**: AES is a symmetric encryption algorithm that is widely recognized and used globally. It encrypts data in blocks, typically of 128 bits, using cryptographic keys of 128, 192, or 256 bits.
* **Hash Functions and Bit Levels**: AES doesn't use hash functions directly but operates at bit levels of 128, 192, or 256 bits.
* **Use of Random Numbers, Symmetric vs. Non-Symmetric Keys**: AES uses symmetric keys, meaning the same key is used for both encryption and decryption. Random numbers (or nonces) can be used in modes like AES-GCM to ensure that the same plaintext doesn't produce the same ciphertext when encrypted multiple times.
* **History and Current State**: AES was established as a standard by the U.S. National Institute of Standards and Technology (NIST) in 2001. It's currently one of the most popular encryption algorithms in use.

## Certificate Generation

Insert a screenshot below of the CER file.

A screenshot of a computer

Description automatically generated

## Deploy Cipher

A screenshot of a computer

Description automatically generated

## Secure Communications

Chrome doesn’t consider selfsigned certificates to be secure, but my webpage forces https as shown below

A screenshot of a phone

Description automatically generated

A screenshot of a computer

Description automatically generated

## Secondary Testing

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

A screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

## Functional Testing

A screen shot of a computer

Description automatically generated

## Summary

**Refactoring and Compliance with Security Testing Protocols**

The goal of this project was to enhance the security of our software application. This process was not just about modifying the code but ensuring that the application aligns with the best security practices and protocols.

**Reference to the Vulnerability Assessment Process Flow Diagram:**

The Vulnerability Assessment Process Flow Diagram serves as a roadmap for identifying and addressing potential security threats. By referring to this diagram, we ensured that our refactoring process was systematic and thorough. Here are the areas we addressed:

1. **Secure Data Transmission**: By implementing HTTPS and ensuring the application uses strong SSL/TLS protocols, we've ensured that data in transit is encrypted and secure from man-in-the-middle attacks.
2. **Dependency Check**: We ensured that all third-party libraries and dependencies are up-to-date and free from known vulnerabilities.

**Process for Adding Layers of Security:**

1. **Code Review**: A systematic review of the codebase was conducted. This wasn't just to find vulnerabilities but also to understand the logic and flow, ensuring that our refactoring wouldn't introduce new issues.
2. **Data Encryption**: Not only did we ensure data in transit was secure, but we also implemented encryption for data at rest. This ensures that even if data is accessed, it remains unintelligible without the decryption key.

In conclusion, our refactoring process was not just about changing the code but ensuring that every change aligns with the best security practices. By following a systematic approach and referring to the Vulnerability Assessment Process Flow Diagram, I’ve ensured that our application is not only functional but also secure against a myriad of threats.

## Industry Standard Best Practices

The application of industry-standard best practices for secure coding is paramount in today's digital age. As cyber threats continue to evolve, adhering to these standards ensures that our software remains resilient against known vulnerabilities and is prepared for future threats.

**Application of Industry Standard Best Practices:**

1. **Code Reviews**: Regular peer code reviews are conducted to ensure that the code not only meets functional requirements but also adheres to security best practices.
2. **Secure Data Transmission**: We enforce the use of HTTPS, ensuring that all data in transit is encrypted.
3. **Logging and Monitoring**: Comprehensive logs are maintained, and real-time monitoring is in place to detect and respond to any suspicious activities immediately.

**Value of Applying Industry Standard Best Practices:**

1. **Operational Continuity**: A secure application ensures uninterrupted business operations. Downtime due to security incidents can lead to significant revenue loss and operational disruptions.
2. **Regulatory Compliance**: Many industries are governed by regulations that mandate specific security practices. Adhering to industry standards ensures that the company remains compliant and avoids potential legal complications.
3. **Future-Proofing**: By following best practices, the software is not only secure against current threats but is also designed with a framework that can adapt to future security challenges.

In conclusion, the application of industry-standard best practices for secure coding is not just a technical necessity but a holistic approach to ensuring the company's operational, financial, and reputational wellbeing. It's an investment in the company's present and future, ensuring that we remain resilient, trustworthy, and ahead of potential threats.