****

# 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 6](#_Toc102040760)

[4. Secure Communications 6](#_Toc102040761)

[5. Secondary Testing 7](#_Toc102040762)

[6. Functional Testing 8](#_Toc102040763)

[7. Summary 9](#_Toc102040764)

[8. Industry Standard Best Practices 9](#_Toc102040765)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **12/6/2022** | **Brylene Patrick** |  |

## 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

Brylene Patrick

## Algorithm Cipher

## Recommend an appropriate encryption algorithm cipher to deploy, given the security vulnerabilities, and justify your reasoning. Review the scenario and the supporting materials to support your recommendation. In your practices for secure software report, be sure to address the following:

## Provide a brief, high-level overview of the encryption algorithm cipher.

## Discuss the hash functions and bit levels of the cipher.

## Explain the use of random numbers, symmetric versus non-symmetric keys, and so on.

## Describe the history and current state of encryption algorithms.

Artemis Financial prioritizes building a strong security system to ensure that their client’s data and financial information is protected. Based on Artemis Financial’s security needs, I recommend SHA-256 as this cipher algorithm provides the most security with low collisions. SHA-256 was designed by the National Security Agency (NSA) with patent from the U.S. government. SHA-256 is recognized as one of the most secure networks globally as their integrity is validated to ensure that security is up to date.

SHA-256 hash function and bit levels are based on randomness and length of encryption. It uses a 256-bit key where a piece of data is taken and converted into a new unrecognizable data string that is fixed length. This network is built with implementation of Merkle–Damgård and Davies-Meyer structure from a specialized block cipher called SHACAL-2 turning the SHA-256 as collision resistant as possible. A random number is used in AES where the output is a result from a large number of bit changes that is distributed from the entire possible range making the secret key more unpredictable. Symmetric key is where ciphers use the same key for encrypting and decrypting. It also involves the conversion of plaintext to ciphertext using the same secret key. Asymmetric key is where two related keys are utilized for encryption. To further elaborate, encryption would be used with a public key while decryption would be used with a private key and vice versa.

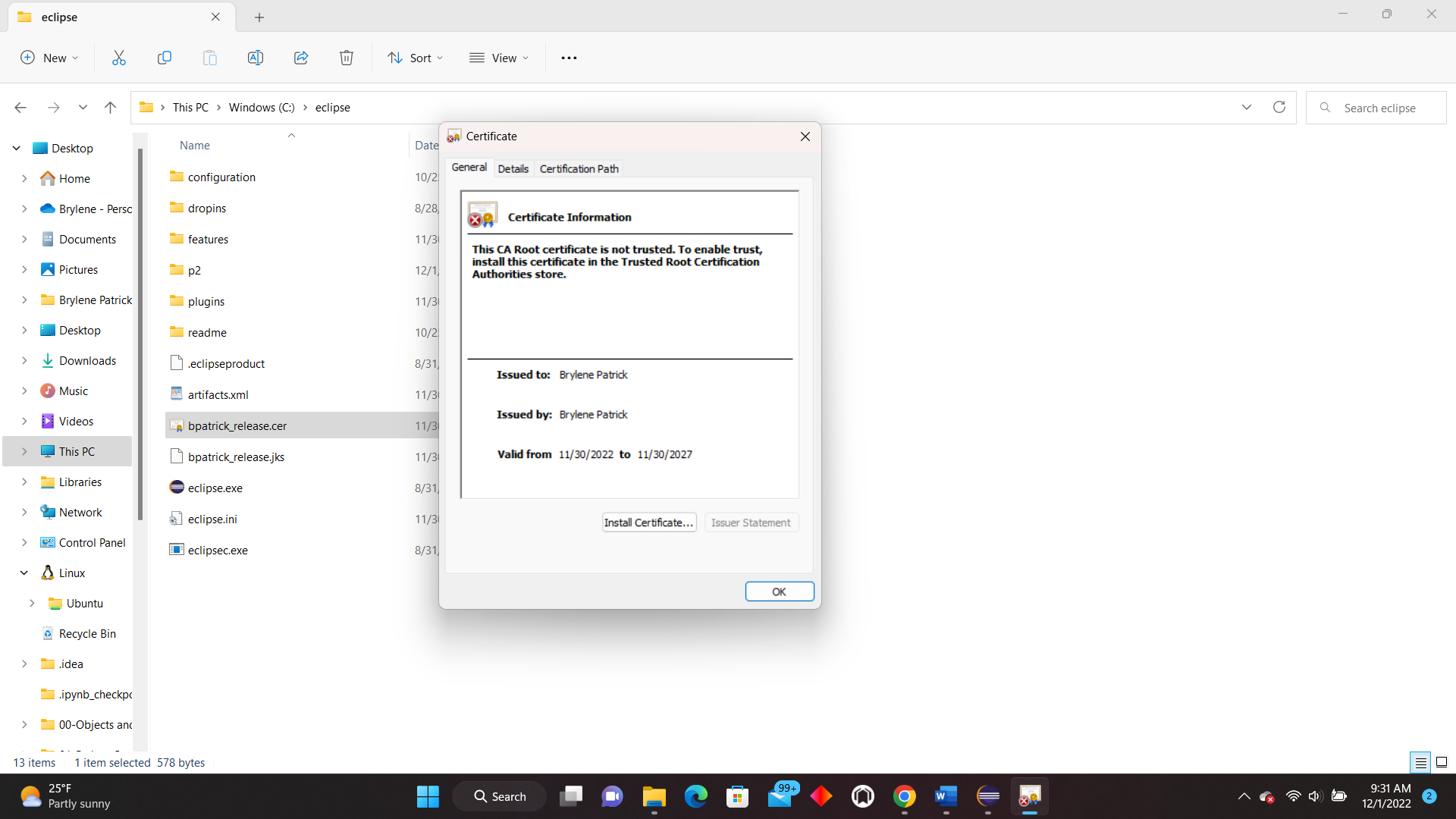
## Certificate Generation

## Generate appropriate self-signed certificates using the Java Keytool in Eclipse.

## To demonstrate that the certificate was correctly generated:

## Export your certificates (CER file).

## Submit a screenshot of the CER file in your practices for secure software report.



Text

Description automatically generated

## Deploy Cipher

Deploy and implement the cryptographic hash algorithm by refactoring code. Demonstrate functionality with a checksum verification.

* Submit a screenshot of the checksum verification in your practices for secure software report. The screenshot must show your name and a unique data string that has been created.

Graphical user interface, text, application, email

Description automatically generated

## Secure Communications

Verify secure communication. In the application.properties file, refactor the code to convert HTTP to the HTTPS protocol. Compile and run the refactored code. Then once the server is running, type https://localhost:8443/hash in a new browser to demonstrate that the secure communication works successfully.

* Create a screenshot of the web browser that shows a secure webpage and include it in your practices for secure software report.

Graphical user interface, text, application

Description automatically generated

A screenshot of a computer

Description automatically generated

## Secondary Testing

## Run a secondary static testing of the refactored code using the OWASP Dependency-Check Maven (see Supporting Materials) to ensure code complies with software security enhancements. You need to focus on only the code you have added as part of the refactoring. Complete the dependency check and review the output to ensure you did not introduce additional security vulnerabilities. Include the following in your practices for secure software report:

## A screenshot of the refactored code executed without errors

## A screenshot of the report of the output from the dependency-check static tester

Screenshot of the refactored code

Graphical user interface, text, application

Description automatically generated

Screenshot of dependency-check report

Graphical user interface, text, application

Description automatically generated

## Functional Testing

## Identify the software application's syntactical, logical, and security vulnerabilities by manually reviewing code.

## Complete this functional testing and include a screenshot of the refactored code, executed without errors, in your practices for secure software report.

screenshot of the refactored code

Graphical user interface, text, application

Description automatically generated

application.properties screenshot

Graphical user interface, text, application

Description automatically generated

pom.xml file screenshot: dependency-check-maven version was updated to 7.3.2 version

Graphical user interface, text, application

Description automatically generated

## Summary

## Discuss how the code has been refactored and how it complies with security testing protocols. In the summary of your practices for secure software report, be sure to address the following:

## Refer to the Vulnerability Assessment Process Flow Diagram. Highlight the areas of security that you addressed by refactoring the code.

## Discuss your process for adding layers of security to the software application.

Areas of security such as cryptography, client/server, and code quality was implemented with security measures to ensure that data is protected. SHA-256 cipher algorithm was used to patent a cryptographic hash function which outputs a value that’s 256 bits long. During this encryption, the data is transformed into an unreadable and secure format that can only be accessed through a private key. The private key ensures a security accessed between the client and server through the incorporation of security certificate. Code quality was enhanced through the addition of a secured RestController and an updated version of the maven dependency check.

The implementation of security certificate was the first step to adding a security layer to the project. A security certificate ensures a reliable source in authenticating and verifying websites for a safe and trusted data that is being accessed by the users. It also notifies users of unsafe websites and other sources of data. SHA-256 was then added for data encryption, website authentication, and secure blockchains. This cipher algorithm also secures communications which is needed considering that Artemis Financials handle international transactions. Lastly, HTTPS enforcement was incorporated into the project so that user access is restricted through an encrypted session. Through the enforcement of HTTPS, communications and data shared between the user and server is further protected.

## Industry Standard Best Practices

## Explain how you applied industry standard best practices for secure coding to mitigate against known security vulnerabilities. Be sure to address the following:

## Explain how you used industry standard best practices to maintain the software application’s current security.

## Explain the value of applying industry standard best practices for secure coding to the company’s overall wellbeing.

Using the industry standard best practices was used to maintain the software application’s current security through the routine dependency checks to detect vulnerabilities. Vulnerabilities are then being carefully analyzed and assessed to navigate the remediations that were taken. In addition, the code quality of the program was reviewed to make sure that it is clear of errors and that the right output data is being delivered. Overall, maintenance of a secure program to address vulnerabilities and other issues includes performing dependency checks on a routine basis and ensuring that security measures are up to date to ensure.