****

# CS 305 Project Two

**Practices for Secure Software Report**

Table of Contents

[Document Revision History 3](#_Toc33111302)

[Client 3](#_Toc33111303)

[Instructions 3](#_Toc33111304)

[Developer 4](#_Toc33111305)

[1. Algorithm Cipher 4](#_Toc33111306)

[2. Certificate Generation 4](#_Toc33111307)

[3. Deploy Cipher 4](#_Toc33111308)

[4. Secure Communications 4](#_Toc33111309)

[5. Secondary Testing 4](#_Toc33111310)

[6. Functional Testing 5](#_Toc33111311)

[7. Summary 5](#_Toc33111312)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **8/9/22** | **Corey Nance** |  |

## Client



## Instructions

Deliver this completed Practices for Secure Software Report documenting your process for writing secure communications and refactoring code that complies with software security testing protocols.

Respond to the steps outlined below and replace the bracketed text with your findings in your own words. If you choose to include images or supporting materials, be sure to insert them throughout.

## Developer

Corey Nance

## 1. Algorithm Cipher

Determine an appropriate encryption algorithm cipher to deploy given the security vulnerabilities, justifying your reasoning. Be sure to address the following:

For Artemis Financial, the recommendation would be AES encryption. It is said that this encryption algorithm is the gold standard and is software and hardware friendly because its immune to all known attacks, its speed and compatibility of source code on many different computing platforms and, its design simplicity (Wallen, 2020).

Hash functions are mathematical functions that map a set of data into a fixed sized string. The bit level is the length of the string or how many bits are in it with 256bit being the strongest, and these functions are a way to ensure data integrity in a public key cryptography (Frankenfield, 2022).

Random numbers are used to inject unpredictable data into cryptographic algorithms and protocols to make sure the streams are unguessable (Manico & Detlefsen, 2015). A symmetric key is when a single key is shared among the people who need to receive the data or message and a non-symmetric key uses a pair of public and private keys. AES is one of the most well-known and effective users of the symmetric key encryption technique (Daniel, 2021).

In the beginning, encryption was a basic technique to of secret writing and hiding messages. Most of it was plaintext or data that was transformed into a ciphertext that prevents outsiders from understanding its content (Team, 2022). With modern encryption, its being used in our lives every day and is still protecting data. The main difference now is the amount of data that is being communicated that needs protection as well as the fact that computers can decrypt faster than humans can. Data has become one of the most valuable commodities in the world and securing that data has become one of the most important tasks in the world.

## 2. Certificate Generation

Generate appropriate self-signed certificates using the Java Keytool, which is used through the command line.

A screenshot of a computer

Description automatically generated

New certificate generated with localhost instead of user name

Graphical user interface, application

Description automatically generated

## 

## 3. Deploy Cipher

Refactor the code and use security libraries to deploy and implement the encryption algorithm cipher to the software application. Verify this additional functionality with a checksum.

Insert a screenshot below of the checksum verification.

Graphical user interface, text, application

Description automatically generated

## 4. Secure Communications

Refactor the code to convert HTTP to the HTTPS protocol. Compile and run the refactored code to verify secure communication by typing **https://localhost:8443/hash** in a new browser window to demonstrate that the secure communication works successfully.

Screenshot of the web browser that shows a secure webpage.

Here the browser is reading not secure due to it being a self-signed certificate.

A screenshot of a computer

Description automatically generated

Here the browser registers as secure but with a warning of certificate being self-signed.A screenshot of a computer

Description automatically generated

## 5. Secondary Testing

Complete a secondary static testing of the refactored code using the dependency check tool to ensure code complies with software security enhancements. You only need to focus on 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.

A screenshot of the refactored code executed without errors

Graphical user interface, text, application

Description automatically generated

A screenshot of the dependency check report

Graphical user interface, text, application

Description automatically generated

## 6. Functional Testing

Identify syntactical, logical, and security vulnerabilities for the software application by manually reviewing code.

Functional testing screenshot of the refactored code executed without errors.

Image of dependency report before suppressions

Graphical user interface, text, application

Description automatically generated

Build success with suppression.xml

Graphical user interface, text, application

Description automatically generated

Dependency report after suppressing false positives

Graphical user interface, text, application

Description automatically generated

## 7. Summary

Discuss how the code has been refactored and how it complies with security testing protocols. Be sure to address the following:

* Refer to the Vulnerability Assessment Process Flow Diagram and highlight the areas of security that you addressed by refactoring the code.
* Discuss your process for adding layers of security to the software application and the value that security adds to the company’s overall wellbeing.
* Point out best practices for maintaining the current security of the software application to your customer.

The project consists of using cryptography, server/client, and code quality. Cryptography is used with generating the secure checksum using 256-bit hashing algorithms to protect the sensitive data. Client/server is used in the set up of having a server send the data to the client side while keeping essential functions like classes and access to a database secure on the server side. Code quality is by following best use coding practices and naming conventions. Also making the code readable and maintainable.

The first layer of security is the SSL certificate for the client side. This certificate is used to check that the site is secure and valid by having a public and private key match. This feature will help customers of Artemis financial to know and trust that their information and data interactions are secure. The second layer is setting the application up so that the application is HTTPS instead of just HTTP. This ensures that the data transfers are also a little more secure and helps to prevent the man in the middle attacks. The final layer is the SHA-256 encryption algorithm. This helps to prevent intruders that may get through from gaining access to sensitive documents. Overall, these layers of security will help Artemis financial keep data secure, keep all transfers secure which in the end will help to gain and maintain user trust and confidence with their interactions with the website.

Best practices for maintaining this application is to schedule dependency checks to make sure there are no new vulnerabilities and to check if any of the false positives have now been updated. Even though client side may have input validators, best practice would be to incorporate input validations on the server side as well for any data coming into the server whether its user input or API data. Lastly would be scheduling code review so that the code can stay up to date against new and upcoming threats.

**References**

Daniel, B. (2021, May 4). Symmetric vs. asymmetric encryption: What's the difference? Trusted Computing Innovator. Retrieved July 21, 2022, from https://www.trentonsystems.com/blog/symmetric-vs-asymmetric-encryption#:~:text=AES%20encryption%2C%20which%20uses%20block,encryption%20techniques%20in%20use%20today.

Frankenfield, J. (2022, July 8). Cryptographic hash functions definition. Investopedia. Retrieved July 21, 2022, from https://www.investopedia.com/news/cryptographic-hash-functions/

Manico, J., & Detlefsen, A. (2015). Iron-clad java building secure web applications. McGraw-Hill Education.

Team, T. (2022, January 31). The history of encryption: The roots of modern-day cyber-security. Tresorit Blog. Retrieved July 21, 2022, from https://tresorit.com/blog/the-history-of-encryption-the-roots-of-modern-day-cyber-security/

Wallen, D. (2020, May 26). *AES encryption: Looking at advanced encryption standards*. Spanning. Retrieved July 20, 2022, from https://spanning.com/blog/aes-encryption/