****

# 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** | **4/16/2023** | **Raymond Sauter** |  |

## 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

Garrett Dunn

## 1. Algorithm Cipher

Determine an appropriate encryption algorithm cipher to deploy given the security vulnerabilities, justifying your reasoning. 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 vs non-symmetric keys, and so on.
* Describe the history and current state of encryption algorithms.

.

Artemis Financial would benefit from AES with SHA-256 hash function as their algorithm cypher. AES is a proven cryptographic cypher it has proven itself against all types of attacks. The one type of attack it has been proven vulnerable to has been Bruce brute force attacks however the likelihood of this is decreased as the bike count becomes more complicated with This cipher. Algorithm is used utilized by many different corporations and platforms to include many government and banking systems. AES uses the same key to encrypt and decrypt data unlike nonsymmetrical keys which create a unique key value for both the encryption and the decryption of the data. The keys generate random numbers at different lengths which makes unauthorized access to the system nye impossible. Encryption algorithms have been around for many thousands of years as they first started to appear during the time of the Greek city states. They were then vastly improved upon in the turn of the 21st century and World War Two with the creation of the enigma machine. Moving forward to today encryption is used in many different areas of your everyday life to include when you access your funds at an ATM machine or make a phone call, and these encryption algorithms get more advanced every single day.

## 2. Certificate Generation

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

* To demonstrate that the keys were effectively generated, export your certificates (CER file) and submit a screenshot of the CER file below.

Text

Description automatically generated

Graphical user interface, text, 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. The screenshot must show your name and a unique data string that has been created.

## 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.

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

Graphical user interface, text, application

Description automatically generated

No matter what I did I couldn’t get it secure, I even added it to the trusted root certification authorities store in mmc. I don’t know what I am doing wrong.

## 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.

* Include the following below:
  + A screenshot of the refactored code executed without errors
  + A screenshot of a computer

    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.

* Complete this functional testing and include a screenshot below of the refactored code executed without errors.
* A screenshot of a computer

  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.

I encrypted the data after code refactoring station by utilizing the hash function at outlined above. This strengthen the security of the API since this is a restful application. To ensure secure data transmission I utilized key tool and keystore to create a certificate. This improved security both on the client side and the server side or the corporate side. I also used a try and catch clause which is an extra layer of added security. I also utilized the dependency checker to make sure both spring boot and tomcat versions we're up to date meaning they have the best security features to safeguard against any known vulnerabilities. All these steps ensure that Artemis financial systems is as secure as possible to safeguard their clients against nefarious acts. Ensuring the system’s security is vital in today's hostile technological climate. With a vast number of security threats making sure your consumer 's data is secure is an important job of the developers nowadays. By constantly monitoring encryption methods, maintaining dependencies, and making sure your code base is error free, are just a few of the methods that you can ensure that no nefarious actors have access to the client’s data.