****

# 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** | **02/20/2022** | **Sorosh Khalili** |  |

## 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

Sorosh Khalili

## 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.

Applications need data for performing some of their tasks, but not every data that the application receives can be trusted and used. Only trusted data are valuable, and the rest must be discarded. After identifying data as trustable, the application must have methods in place to protect the integrity of trusted data when transferring or storing them. This ensures the security of the application and its trusted data and prevents unauthorized access to them in the event of any data breach. Therefore, data protection becomes very important. Regardless of the nature of data (being in transit or at rest), developers must protect its integrity. Data encryption is one of the common ways of protecting data. Developers utilize data encryption algorithms (to convert data into a code) to ensure the safety and security of in-transit and at-rest data. But what if the application needs to use the data. Its encrypted form is unusable to the application, so it must utilize data decryption algorithms (to convert data back to its original form) to use the data. However, cipher algorithms (instruction set or method to encrypt and decrypt the data) are not designed equally in their protection level, performance, and ability to withstand an attack.

Advanced Encryption Standard (AES) is an excellent cipher algorithm approved by the National Institute of Standards and Technology (NIST). The United States government's goal in creating Federal Information Processing Standard (FIPS) for encryption with a high protection level capable of safeguarding sensitive government information has led to the design and development of the AES. The AES uses the block size of 128 bits which can have one of the three different key length choices of 128, 192, and 256 bits. The AES’s unique characteristics make it an appropriate cipher algorithm for encrypting Artemis Financial archive files. The 128-, 192-, and 256-bits key sizes use 10, 12, and 14 rounds to generate the key respectively. Hence, a higher length key size utilization makes it very secure. As a result, it becomes almost impossible to break this encryption method.

The AES encryption process starts by dividing the data into blocks. Then the first round of key expansion takes place on the second step of encryption. After expanding the key, the cipher algorithm adds the initial key and the block of data together. For the next step, the algorithm starts to substitute every byte with a code based on the pre-defined table called the Rijndael S-box. Then rows of the block will be shifted. After finishing this step, a predefined matrix will be used to mix the columns and create a new block of code. By this time, the algorithm adds the previous expanded key and the newly created block together. Finally, the AES algorithm repeats the process until it reaches the total number of rounds based on the initial number of the chosen key bit.

The AES algorithm still has its advantages and disadvantages, despite being recommended to encrypt Artemis Financial archive files. It is certainly very difficult to identify a cipher algorithm as a perfect algorithm because they have advantages and disadvantages. The AES also has a set of advantages and disadvantages. The strengths of the AES algorithm are the ability to be implemented on both hardware and software, providing a high level of security, and it is open source. The weaknesses of the AES algorithm are requiring multiple rounds of encryption that can potentially reduce the speed and performance and lots of processing and calculation in each different stage.

Hash functions are very common, and they are almost present in all cipher algorithms. A hash function is a mathematical process and holds a critical role in public-key cryptography. These mathematical functions transform an input set of data into a fixed-size bit string that is called the hash value. These functions help to store passwords securely in the database, authenticate a user, ensure the integrity of the data, and efficiently organize files and other contents. Simply put, a hash function converts any size data input into a unique ciphertext with a fixed length of bits. The hash function is a unique identifier for each specific data which means function changes based on the data. The purpose of encrypting data is to protect information during its transmission. The number of bits the cipher algorithm uses has a direct impact on its level of strength. Although a 128-bit cipher is secure and difficult to break, the 256-bit cipher is significantly stronger and provides better protection.

In cipher algorithms, the ability to generate random numbers is very important, because it prevents prediction based on reasoning and logic. Cybercriminals often try to gain unauthorized access based on clues in the application’s behavior. Therefore, random number generation becomes an effective defense mechanism in cipher algorithm implementation. Symmetric key encryption is the simplest encryption and uses only one secret key to encrypt and decrypt the data. Having one secret key makes symmetric encryption easy to use but reduces its security. Therefore, transferring its secret key safely between the involved parties is very important. The symmetric encryption algorithm is the favored choice for transmitting large amounts of data since it has less complexity and executes faster. The non-symmetric (asymmetric) key encryption utilizes two different keys for encryption and decryption called private and public keys. Using two different keys makes non-symmetric key encryption more secure but much slower.

Cryptography is the process of concealing information from unauthorized access or view. The earliest documented encryption model substituted data with non-standard symbolic representations to conceal them and then used individual recognizable proof checks such as logo, seal, or insignia for its verification. The recipient had a duplicate of the genuine imprint to verify the originality of the information. Currently Advanced Encryption Standard (AES) is the most used and recognizable method of encryption for securing sensitive information on the web. The constant technological advancements and increase in speed and processing power of connected devices have made it possible to utilize longer keys to ensure higher security levels against constant threats on the internet.

## 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

## 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.

Graphical user interface, text, application, email

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.

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

Graphical user interface, text, application

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.

* Include the following below:
  + A screenshot of the refactored code executed without errors
  + A screenshot of the dependency check report

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, email

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.

Graphical user interface, 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.

Areas of security:

* APIs: Secure API Interactions
* Cryptography: Encryption Use and Vulnerabilities
* Client / Server: Secure Distributed Composing
* Code Quality: Secure Coding Practices / Patterns

Having a secure API ensures safe and secure interaction between the user and the system. Therefore, it is important to inspect and evaluate the program code to ensure that API provides secure interaction. This application requires access to sensitive customer information, also the customer’s web browser needs to interact with the server to perform requested tasks. Sensitive information must be encrypted, and a secure connection is required to transmit this data between the client and the server. As a result, the client/server and the cryptography must be present and work side by side to protect the customer’s security. The self-signed certificate authority has been utilized to test and ensure the proper functionality of the application. For the data encryption, the AES-128 has been utilized as the cipher algorithm. The vulnerabilities put the application at risk, therefore, producing high-quality code is very important. Also, it is crucial to run the dependency check as well as manually test the program codes to ensure its security.