****

# CS 305 Project Two

**Practices for Secure Software Report**

Table of Contents

[CS 305 Project Two 1](#_Toc96289537)

[Document Revision History 3](#_Toc96289538)

[Client 3](#_Toc96289539)

[Developer 4](#_Toc96289540)

[1. Algorithm Cipher 4](#_Toc96289541)

[2. Certificate Generation 4](#_Toc96289542)

[3. Deploy Cipher 5](#_Toc96289543)

[4. Secure Communications 5](#_Toc96289544)

[5. Secondary Testing 6](#_Toc96289545)

[6. Functional Testing 7](#_Toc96289546)

[7. Summary 8](#_Toc96289547)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **2/20/2022** | **Kareem Elkwae** | **Completed document** |

## Client



## Developer

Kareem Elkwae

## 1. Algorithm Cipher

Artemis Financial would like to be able to verify the integrity of their files by generating a checksum. The algorithm cipher that I would recommend is the SHA-256 algorithm cipher. This cipher is a one way, meaning the cipher will produce a 256 bit hash, and the information contained within the hash cannot be discerned from the hash itself. Putting the same input into the cipher will always produce the same hash, so it can be used to see if the input has been altered, which is why is it appropriate for the needs of Artemis Financial.

The SHA-256 Algorithm works by taking the input, converting it to binary, appending the binary to a certain length, then performing mathematical operations on the binary until a 256 bit output is produced. There are varying levels of bit length for the SHA family of algorithms. We have selected 256 bit because of its relatively efficient way of producing hashes that do not contain collisions. We could use a larger bit value version, but it would decrease performance and is not necessary.

In this algorithm, no random numbers are used other than the 0’s appended to the end of the binary version of the file being hashed to bring it to the proper length. The SHA-256 algorithm is asymmetric, which means that the algorithm does not work the same to create the hash as it would to discern the content from the hash. This is especially useful since it is virtually impossible to reproduce the original content from the hash. A symmetric algorithm would allow for us to run content through the cipher to generate a hash, and then we would also be able to run the hash back through the cipher to reproduce the original content. A symmetric algorithm to secure the communication between client and server.

In the past, encryption algorithms have been used in a variety of ways. Most commonly, encryption is used to store and transmit data securely. It is also used to verify file integrity. It is still used for the same purposes today, but the methods of encryption have been updated as flaws in the methods have been identified. Some forms of encryption have been abandoned as new and improved methods have been created. Because digital security is so important in the modern world, there are institutions dedicated to evaluating best security practices and collecting information about various security vulnerabilities.

## Graphical user interface, text, application Description automatically generatedText Description automatically generated2. Certificate Generation

## 3. Deploy Cipher

Graphical user interface, text, application, email

Description automatically generated

## 4. Secure Communications

Graphical user interface, text, application

Description automatically generated

## Graphical user interface, text Description automatically generatedGraphical user interface, text, application, email Description automatically generatedGraphical user interface, text Description automatically generated5. Secondary Testing

## 6. Functional Testing

All syntactical, logical and security vulnerabilities derived from the implementation of the cipher and the certificate are addressed. The only remaining vulnerabilities are from the dependencies that are in use and from the password being saved in a cleartext format in the application.properties file.

Graphical user interface, text, application

Description automatically generated

## 7. Summary

Vulnerability Assessments:

1. API: The implementation of HTTPS and ensuring that a user would be able to connect and interact with the server through the use of various maven dependencies demonstrates secure API interactions.
2. Cryptography: The implementation of a SHA-256 algorithm cipher demonstrates the use of encryption in the system to verify the integrity of information.
3. Client/Server: The use of a certificate authority to enable a secure connection to the server demonstrates the ability of the system to maintain a secure client/server relationship.
4. Code Error: The one main error that could come up, the “NoSuchAlgorithmException” is handled through the use of the java dependency. The rest of the code is without error.
5. Code Quality: The code has been reviewed to ensure that is functions properly and efficiently.

One way that we have added to the security of the software application is by creating and implementing a secure self-signed certificate to validate whether or not the web address we are visiting is truly on the server. We also ran a dependency check before and after implementing the additional code to ensure that no new vulnerabilities were created by the code we added. The other layer of security that has been added to this software is the encryption algorithm. We implemented the SHA-256 algorithm cipher which allows us to validate the integrity of data in the system. We can expand the algorithm from what we have already implemented to include all files within the system.

Some best practices for maintaining the current security of the software application would be to continue to keep up to date on newly discovered vulnerabilities. We can do this by running dependency checks. We must also be sure to implement whatever patches/fixes are available for any vulnerabilities found. We must also make sure to limit access to the system and its components to only trusted individuals. We can do this by implementing some sort of credentialed system access policies.