****

# 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 4](#_Toc102040760)

[4. Secure Communications 4](#_Toc102040761)

[5. Secondary Testing 4](#_Toc102040762)

[6. Functional Testing 4](#_Toc102040763)

[7. Summary 4](#_Toc102040764)

[8. Industry Standard Best Practices 4](#_Toc102040765)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **10/21/2023** | **Tim OHagan** |  |

## Client



## Developer

[Tim OHagan.]

## Algorithm Cipher

An encryption algorithm cipher is a mathematical algorithm that is used to encrypt and decrypt data. Ciphers use a key to transform plain text into ciphertext and ciphertext back into plain text. The key is a secret piece of information that is known only to the sender and receiver of the data. I recommend that Artemis Financial use AES to encrypt its sensitive data. AES is a highly secure encryption algorithm that is used by financial institutions around the world. It is also relatively easy to implement and use.

Encryption algorithms have been used for centuries to protect sensitive data. Some of the earliest encryption algorithms were simple substitution ciphers, which replaced letters in the plain text with different letters in the ciphertext. More modern encryption algorithms are much more complex and difficult to break. Advanced Encryption Standard (AES) is one of the most widely used encryption algorithms today. AES is a symmetric key encryption algorithm that uses a 128-bit, 192-bit, or 256-bit key to encrypt and decrypt data. AES is considered to be one of the most secure encryption algorithms available. It has been extensively tested and reviewed by cryptographers, and no known security vulnerabilities have been discovered.

A hash function is a cryptographic algorithm that converts data of any size into a fixed-size hash value. Hash values are unique for each input data, and it is very difficult to find two different input data that produce the same hash value.

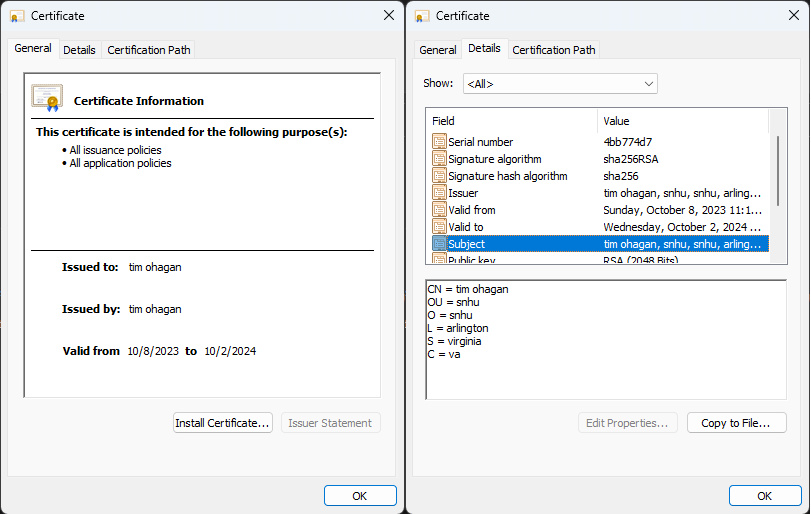
The bit level of an encryption algorithm cipher refers to the length of the key that is used to encrypt and decrypt data. The longer the key length, the more secure the encryption is. However, longer key lengths also require more computational power to encrypt and decrypt data.

Symmetric key encryption algorithms use the same key to encrypt and decrypt data. Non-symmetric key encryption algorithms use two different keys: a public key for encryption and a private key for decryption.

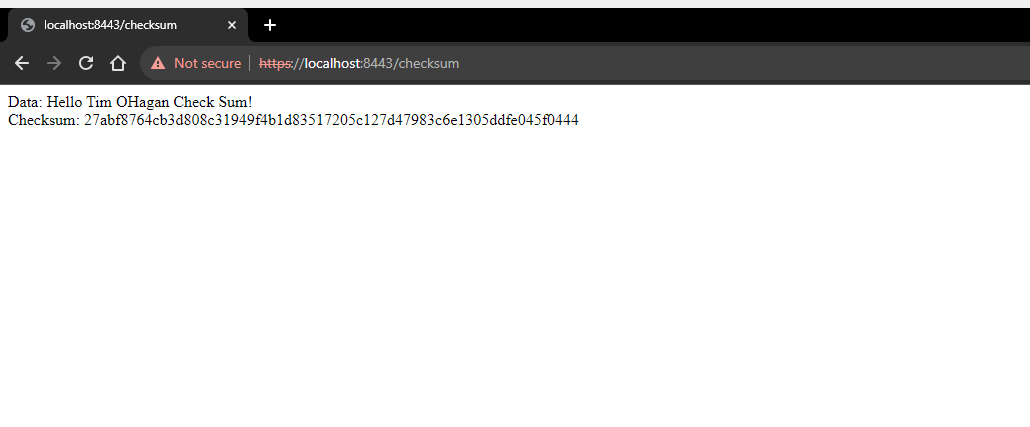
Symmetric key encryption algorithms are generally faster than non-symmetric key encryption algorithms. However, symmetric key encryption algorithms require both the sender and receiver to have the same key. This can be a problem for secure communication over public networks.

Non-symmetric key encryption algorithms are slower than symmetric key encryption algorithms. However, non-symmetric key encryption algorithms do not require the sender and receiver to have the same key. This makes non-symmetric key encryption algorithms ideal for secure communication over public networks.

## Certificate Generation



## Deploy Cipher



A computer screen shot of a program

Description automatically generated

## Secure Communications

A screenshot of a computer

Description automatically generated

## Secondary Testing

A screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

## Functional Testing

Input validation: The generateChecksum() method does not perform any input validation on the data parameter. This could allow an attacker to inject malicious code into the checksum generation process, which could then be executed when the checksum is verified.

Hardcoded key: The generateChecksum() method does not use a secret key to generate checksums. This means that an attacker could potentially generate their own checksums and use them to forge requests or data.

A screenshot of a computer program

Description automatically generated

## Summary

The following security measures were implemented in the web application:

HTTPS: HTTPS was implemented to encrypt all communication between the client and server. This prevents attackers from eavesdropping on or tampering with data in transit.

Browser interface: A browser interface was created for users to view the application. This interface is more secure than a direct API interface, as it is less susceptible to attack.

Encryption algorithms, ciphers, and hash functions: Encryption algorithms, ciphers, and hash functions were used to encrypt sensitive data and verify checksums. This helps to protect data from unauthorized access and modification.

Checksum verification: Checksum verification was implemented to ensure that data has not been tampered with in transit.

Client/Server architecture: The application was designed using a client/Server architecture. This helps to isolate the application from potential attacks.

Code error handling: Exceptions were used to handle code errors. This helps to prevent the application from crashing if an error occurs.

Code review: The code was reviewed to ensure functionality and readability. This helps to identify and fix potential security vulnerabilities.

Self-signed certificates: Self-signed certificates were used to enable HTTPS. This provides a basic level of security, even though the certificates are not trusted by default.

Dependency check: The pom.xml file was refactored to resolve all vulnerabilities discovered in the dependency check. This helps to ensure that the application is not vulnerable to known attacks.

## Industry Standard Best Practices

In addition to the security measures already implemented, there are a number of best practices that can be followed to maintain the security of the web application, including:

Patch software and systems: It is important to patch software and systems regularly to ensure that they are up to date and protected from the latest vulnerabilities.

Enforce least privilege: Users should only have the access that they need to perform their job duties. This helps to prevent attackers from gaining access to sensitive data or systems if they are able to compromise a user account.

Use a firewall: A firewall can be used to block unauthorized access to the application.

Monitor logs: Logs should be monitored for suspicious activity. This can help to identify and respond to attacks early on.