****

# 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** | **12/8/2023** | **Coty Mugford** |  |

## Client



## Instructions

Submit this completed practices for secure software report. Replace the bracketed text with the relevant information. You must document your process for writing secure communications and refactoring code that complies with software security testing protocols.

* Respond to the steps outlined below and include your findings.
* Respond using your own words. You may also choose to include images or supporting materials. If you include them, make certain to insert them in all the relevant locations in the document.
* Refer to the Project Two Guidelines and Rubric for more detailed instructions about each section of the template.

## Developer

Coty Mugford

## Algorithm Cipher

Generally, I would choose to implement the Advanced Encryption Standard (AES). The reason I would choose AES is because it has become the standard in encryption as it is nearly impossible to crack. Banks use it, educational institutions use it, and government agencies use it. It Is the recommended cipher by NIST. On top of that, at this point where our technology is, even at the 256-bit level, it is still a highly efficient encryption algorithm, which is important as well. AES does have a built-in hash function called AES-Hash that will work similarly to the SHA hash function. AES is a symmetric encryption algorithm. However, the recommended hash function by NIST is SHA1 and SHA2 Family of hash algorithms.

Because of the recommended hash function being SHA 256 by NIST, I would choose to pair it with RSA (Rivest-Shamir-Adleman) encryption. The reason I would pair these together is because RSA is an asymmetric algorithm. This would prove to be very useful for Artemis Financial as they will have many users that will need their data encrypted. With RSA being asymmetric, you would be able to provide one key to all your users and a separate encryption key to decode it for each user. That is one of the strengths of asymmetric encryption.

The difference between symmetric and asymmetric keys is that symmetric keys use the same key for all users. This means that keys that encrypt and decrypt messages must remain secret to ensure privacy. An asymmetric key is when one key is made public so that anyone who needs it can have it, and the other key is kept secret. These separate keys are tied together via mathematical logic and can only be decrypted using the other key and nothing else.

Encryption as a concept has been around for a very long time, dating back to ancient Sparta where they used a rod and leather strap to encrypt messages. They would do this by encrypting a message on the leather strap and only using a specifically sized rod could you decrypt the message. Encryption would continue to be used in this way and in other similar ways up until 1917 where the Hebern Rotor Machine was invented and thins consisted of a rotating disc that would embed a key into it and would encode a substitution that would change every time a new character was typed. This and other similar methods were used through World War 2. After World War 2, in 1945, Claude E. Shannon of Bell Labs published an article called “A Mathematical theory of cryptography.”. This would become the starting point to modern cryptography as we know it today. Up until 1970, these modern techniques of encryption were only used by the government. After 1970, IBM formed a cryptography group that designed a block cipher to protect its customers which was later adopted by the US government as the standard encryption cipher. This would become the Data Encryption Standard or DES. This remained the standard until 1997 when it was relatively easily cracked. In 2000, AES was developed and has been the standard by the US government ever since. It is still considered uncrackable today.

## Certificate Generation

A computer screen with white text

Description automatically generatedInsert a screenshot below of the CER file.

## Deploy Cipher

Insert a screenshot below of the checksum verification.

A screenshot of a computer

Description automatically generated

## Secure Communications

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

A screenshot of a computer

Description automatically generated

## Secondary Testing

Insert screenshots below of the refactored code executed without errors and the dependency-check report.

A screenshot of a computer

Description automatically generatedA screen shot of a computer

Description automatically generated

I believe that the vulnerabilities shown above are those from using outdated versions. I do not believe any of them are due to the refactored code. Moving on to functional testing.

## Functional Testing

Insert a screenshot below of the refactored code executed without errors.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

## Summary

The way that the code has been refactored was mainly using the Vulnerability Assessment Process Flow diagram. To start, I went through my code and checked to make sure that I was properly validating input and securing variables. From there, I moved on to ensure that my encryption protocols were properly functioning. Since they were functioning properly, I ensured that the encryption between the server and the client was functioning. I did this by securing the port, upgrading the server’s original http to https. This was done by self-signing a security certificate for the server. Next, I made sure that I had proper error handling, and I did this by implementing a try/catch statement. I used a NoSuchAlgorithm catch to ensure that if the algorithm is not properly functioning, the error will be handled. Finally, I ensured that I had proper code quality and encapsulation. I did this by ensuring that my variables were properly set to private or final if need be.

## Industry Standard Best Practices

In this project, I used industry best practices by using a suitable encryption algorithm, in this case RSA. Alongside it, I used a powerful hash function, namely SHA-256. Both are extremely effective at their respective jobs and would supply sufficient security for Artemis Financial. I also secured the connection to the server by creating a self-signed certificate, changing it from http to https. Alongside that, I ensured secure code by following the Vulnerability Assessment Process Flow chart.

Once all that was said and done, I ran several tests to ensure that the program was indeed secure. I started by checking that the connection was secure by connecting to the port and seeing that it was indeed a https connection. From there I also was able to see that the original text was encrypted. Next, I did secondary testing by using the OWASP dependency-check. This allowed me to see known vulnerabilities in my code. After ensuring that any vulnerabilities introduced by me and my code were gone, I did one final test by going through my code and manually checking for errors. Through all 3 phases, I was able to come up with a secure program that works as it is supposed to.

Ensuring that the industry’s best standards are used is extremely important with any subject, but it is even more so with security. Nearly everyone is on the internet these days and more and more of our personal information is used on the internet. Alongside that, it is becoming easier and easier to learn how to hack, even for the average person who has next to no computer experience. This means that there are more and more bad actors out there every day, searching for people’s highly valuable information. This means that it is crucial for developers to be on their A-game when it comes to securing their programs. It can have horrible consequences for both the user and the service provider. When people’s information gets stolen from a company that a user thought they could trust that trust is lost and you will have probably lost that customer but also all their friends and family as well. If this happens on a grander scale, this can have devastating consequences for the company.

## Bibliography

* Arcserve. (2023, September 19). 5 common encryption algorithms and the unbreakable of the future. https://www.arcserve.com/blog/5-common-encryption-algorithms-and-unbreakables-future
* A brief history of encryption (and cryptography). Thales Group. (2023, February 1). <https://www.thalesgroup.com/en/markets/digital-identity-and-security/magazine/brief-history-encryption>
* Manico, J., & Detlefsen, A. (2014). Iron-clad java: Building secure web applications. McGraw-Hill Education.