****

# 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** | **10/12/2022** | **Steven Colley** |  |

## 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

Steven Colley

## Algorithm Cipher

## Encryption is used to protect data while in transit for both the sender and receiver. The data is encrypted using an algorithm cipher that sends the message throw a mathematical formula and is only solvable with a key. These keys can be authenticated by a certificate authenticator. These are used to validate if the sender and receiver are who they say they are. Once the data is received the data can be sent through a cipher like the one used within the assignment, SHA-256, to decrypt the message. SHA-256 is a hash function that was created by the National Security Agency (NSA). It is a 256-bit hash function that allows for secured encryption while not having to give up on performance. There is technically a stronger function of SHA-512 but it’s much more resource intensive.

## A random number generator is used to help with the encryption by not allowing for any patterns while encrypting data. The are send through hash functions and come out gibberish to only be made understandable but the cipher and appropriate key. The key is handled in two different ways that being symmetric and non-symmetric keys. The symmetric key is a system is which a single key is used for ciphering and deciphering of data the sender sends the key privately to unlock the hidden message. A non-symmetric key is a combination of two different keys being used. The sender and receiver have a public key that is sent to one another while they also have a private key that is sent. As an example, the key would be a number being “X” for the sender. While the receiver has a key of “Y” that is unknown to the public. This is multiplied to create a unique key “Z” that is unstillable unless all keys are obtained.

## Encrypting data has been around for years with the most famous device being the enigma machine that was used during WWII. This was used by the German forces to protect their messages. The only ones that could read a German encrypted messages were those who had an enigma machine and know a specific code that was used during the use of the machine, both for writing the message and reading a message. This system is very similar to what we use in computers every day, but the information is much more secure because I number of ciphers can be used at anytime and less chances of human errors.

## Certificate Generation

Insert a screenshot below of the CER file.

Text

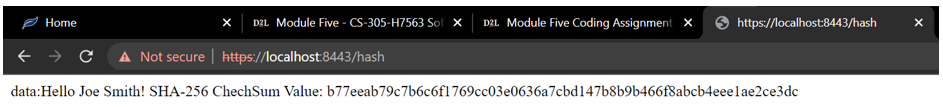
Description automatically generated

## Deploy Cipher

## Text Description automatically generated

## Secure Communications

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



## Secondary Testing

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

Text

Description automatically generated

Text

Description automatically generated

## Functional Testing

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

Text

Description automatically generated

## Summary

Listed in the assignment we were asked to include the code that would use a checksum which allows for data validation. This is used for making sure all data has been received properly and unaltered by a possible man in the middle attack. We were also asked to add in a self-certificate signature that ensures the receiver that the data is coming from a trusted source and would also allow for the use of https formatted websites.

To determine what is the best uses for the different levels of security the answer must be asked what is being sent and how important is it to be secured. We first look at how the data being sent and what dependencies will be in play for it. Then follow through the flow diagram and assess if each stage is necessary for our sending of data. Finally incorporate the security measures needed.

## Industry Standard Best Practices

The assignment had use test for dependencies and checking what could be at risk while sending information. We were to review those dependencies and add in security measures that would protect us in the future like incorporating a signature, checksum and encrypting data using a cipher. We were given a flow diagram of the different levels of security that could help us with a type of checklist to user nothing was missed. These practices are very important in the efforts of maintaining a secured experience for us but those who we develop code for. The dependency check ensures that no security measure is overlook or forgotten. It is also important for the overall health of the company because customers trust us with their data and to ensure no foul play is occurring. Once that trust is broken with a security breach it may never be given again.