**CS 305: Project Two**

Chris Austin

CS-305-T2632 Software Security 21EW2

Dr. Vivian Lyon

December 12, 2021

****

# 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** | **December 12, 2021** | **Chris Austin** |  |

## 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

Chris Austin

## Algorithm Cipher

We employ encryption to prevent arbitrary access to data. Ciphers are algorithms used to perform encryption and hashing (Manico & Detlefsen, 2015). Ciphers take input data and then jumbles the data so that it will not be understandable for someone who hasn’t been given a key to decrypt the data. There are several items or concepts that are important to how ciphers function including bit levels, hash functions and, random numbers. Furthermore, there are two common types of encryptions known as symmetric and non-symmetric encryption. Encryption has long history which has led to modern cipher and hash functions.

Hash functions are mathematical algorithms that map any length of data to a fixed sized with an idempotent result (Mehta, 2020). Checksums use this feature of the algorithm to verify the content of files, folders and just about anything else in a deterministic manner. But, not all hash functions are appropriate for the task at hand.

Many ciphers will be named with an algorithm name followed by a hyphen and a bit level. For example ***sha-256***. The number after the hash function is know as the bit level. A bit is smallest piece of information that can be represented by a computer. The bit-level in a hash function is the number of bits the has function will produce (Mehta, 2020).

Random numbers are used in many ciphers. To prevent an attacker from guessing encryption keys, random numbers are used when they are created (Mehta, 2020). When the same encryption key for encryption and decryption this is symmetric encryption.

The method should be employed when all users that require to access the encrypted data are granted access to the encryption keys (*What is asymmetric encryption?*). But, in the case that encrypted data is shared by a sender to a receiver to be decrypted later with a different encryption key asymmetric encryption is employed. Asymmetric encryption is often used for communications where keys are exchanged beforehand. See SSL and HTTPS as examples of applications of asymmetric encryption (*What is asymmetric encryption?*).

There are weak hash functions like SHA-1 and MD5 that suffer from being vulnerable to collisions. When a has function is vulnerable to collisions, an attacker can construct forged input that will have the same output as a original data (Mehta, 2020). SHA-256 and SHA-512 are known to resist collisions (Mehta, 2020).

Because we are trying securely upload files, we should takes steps to ensure that the uploaded items haven’t been tampered with. Therefore, a cryptographic hash function must be used to generate a verifiable signature of each file. For this task, we are recommending a SHA-256 hash function.

## 2. Certificate Generation

The generated certificate.

Text

Description automatically generated

## 3. Deploy Cipher

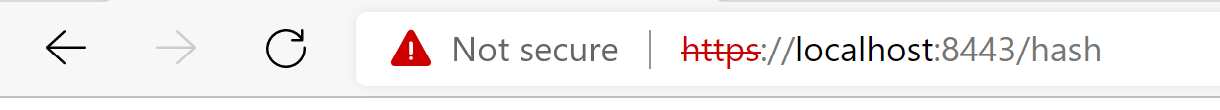
Below is an example of the functioning cipher.

Graphical user interface, text, application, email

Description automatically generated

## 4. Secure Communications

Below is an image of https being employed with a self signed certificate.



## 5. Secondary Testing

The following is a screenshot of the changes to the pom file to ensure secure dependencies.

Graphical user interface, text, application

Description automatically generated

The following is a screenshot of the dependency check report.

Graphical user interface, text, application, email

Description automatically generated

## 7. Summary

## This project has been refactored to use SSL and the latest secure version of it’s core dependency spring boot. By using SSL we are ensuring that asynchronous encryption is used for all communication with the server. By updating springboot to the latest version we are eliminating insecure dependencies and false positives.

## This web application has the following area of concerns.

* Input Validation: Because we will be accepting file uploads, we should ensure that those files are validated.
* APIs: This application is accessed via an API. So, we must secure those APIs with access controls.
* Cryptography: We app should use secure communication vial SSL/TLS.

The process for adding layers of security to the application should start with monitoring dependencies ensuring that vulnerable packages are patched. Additionally, manual code reviews should take place with every change to make sure that no new programmer related issues are introduced. The best way to ensure this is to automate as much as reasonable via continuous integration tooling.

References:

Mehta, M. (2020, December 4). *Hash function in cryptography: How does it work?* InfoSec Insights. Retrieved November 22, 2021, from https://sectigostore.com/blog/hash-function-in-cryptography-how-does-it-work/.

*AES 256 encryption in Java*. Javatpoint. (n.d.). Retrieved November 22, 2021, from https://www.javatpoint.com/aes-256-encryption-in-java.

Manico, J., & Detlefsen, A. (2015). *Iron-clad java: Building secure web applications ; best practices for secure Java Web Application Development*. Oracle Press.

Mehta, M. (2020, December 4). *Hash function in cryptography: How does it work?* InfoSec Insights. Retrieved November 22, 2021, from https://sectigostore.com/blog/hash-function-in-cryptography-how-does-it-work/.

*What is asymmetric encryption? | asymmetric ... - cloudflare*. cloudflair. (n.d.). Retrieved November 22, 2021, from <https://www.cloudflare.com/learning/ssl/what-is-asymmetric-encryption/>.

Crawford, D. (2019, February 4). *AES encryption: Everything you need to know about AES*. ProPrivacy.com. Retrieved November 22, 2021, from https://proprivacy.com/guides/aes-encryption.