GLOBALRAIN

Practices for Secure Software Report

Table of Contents

Document Revision History

Version	Date	Author	Comments
1.0	21 April 2024	Bryan Chan	In this report, I recommend the SHA-2 encryption algorithm for our clients. In addition, I refactored the provided code base, and implemented a self-signed certificate. After this was done, static and functional testing was one in order to ensure that the codebase contained no vulnerabilities.

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
Bryan Chan

1. Algorithm Cipher

For the purposes of distributing public keys to our clients, the best encryption algorithm cipher that will do the best job in avoiding collisions is the SHA-2 algorithm. The SHA-2 algorithm is a secure encryption algorithm that is used to secure user data from attackers. According to the Encryption Consulting article, "What is SHA? What is SHA used for" (2023), "SHA-2 can produce a variety of bit-lengths, from 256 to 512 bit, allowing it to assign completely unique values to every hash digest created" (Encryption Consulting, 2023). This means that the SHA-2 algorithm specializes in creating unique hash values from data using large amounts of bit-lengths. In addition to this, the SHA-2 algorithm also results in the least amount of collisions when compared to other encryption algorithms. For instance, according to the previous Encryption Consulting article (2023), "If SHA-2 is used, there will likely be few to no collisions" (Encryption Consulting, 2023).

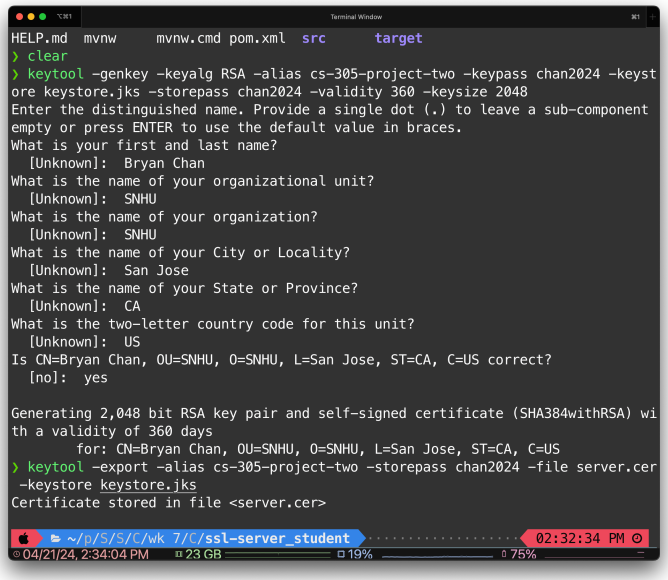
Hash functions are also used in encryption algorithms to protect input data. According to Medha Mehta in the InfoSec Insights article, "Hash Function in Cryptography: How Does It Work?" (2020), a hash function takes the input data and turns it into a unique and irreversible output value (Medha, 2020). Hash functions are also important since they secure data by ensuring that only the original account owner has access to the data. Medha Mehta in the previous article (2020) makes the observation that the bit levels used by the hashing function is chosen based on the length of the output or the type of algorithm being used (Mehta, 2020)

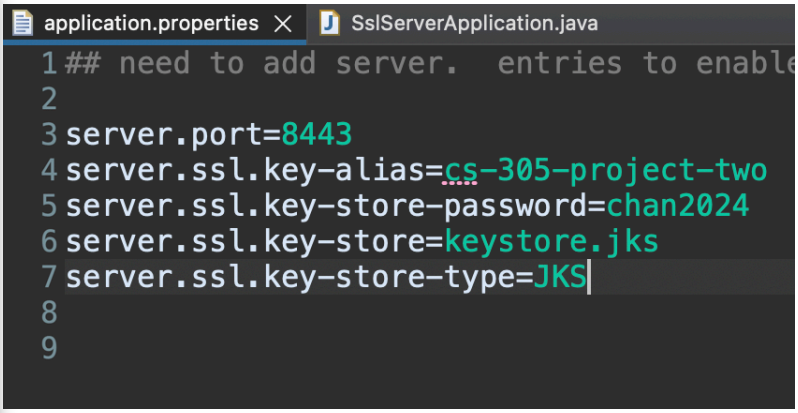
Hashing functions also use random numbers. For instance, Medha Mehta in the previous InfoSec Insights article (2020), states that random characters are added to input values before hashing in order to make hashed values more secure (Mehta, 2020). Random numbers offer more protection to input values that have been hashed.

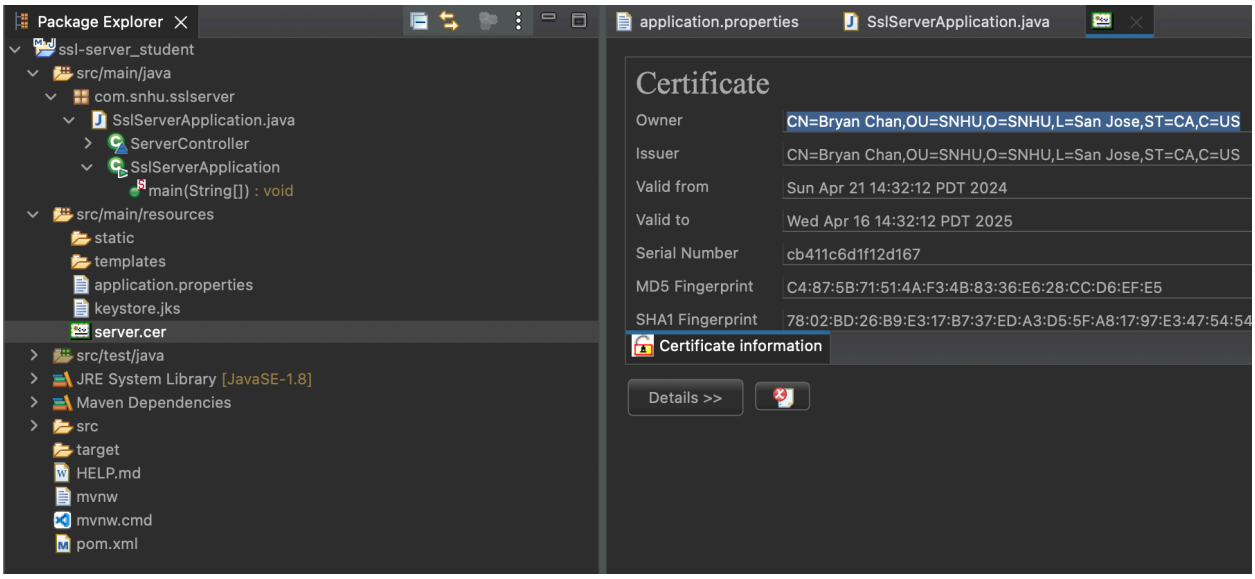
Besides random numbers, hash functions also utilize symmetric and asymmetric keys. Medha Mehta (2020) notes that symmetric encryptions keys can decrypt data using the same key, while asymmetric keys decrypts data using a different key (Mehta, 2020). The distinction in encryption keys offers Artemis Financial options when it comes to dealing with hashed data.

Encryption algorithms are nothing brand new. For example, the Thales Group article, "A brief history of encryption (and cryptography)" (2023), mentions that "[…] [the] ancient Spartans used a scytale device to send secret messages during battle." (Thales Group, 2023). However, in today's world complex encryption algorithms are used by companies to secure message transmissions, financial transactions, and other transfers of data. In the context of this project, the SHA-256 algorithm was implemented in order to encrypt all incoming data with the least amount of collisions possible.

2. Certificate Generation



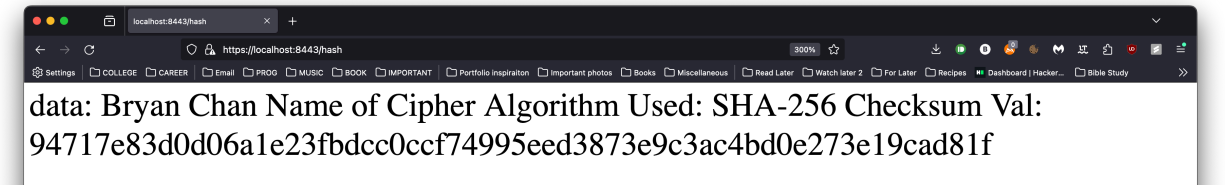




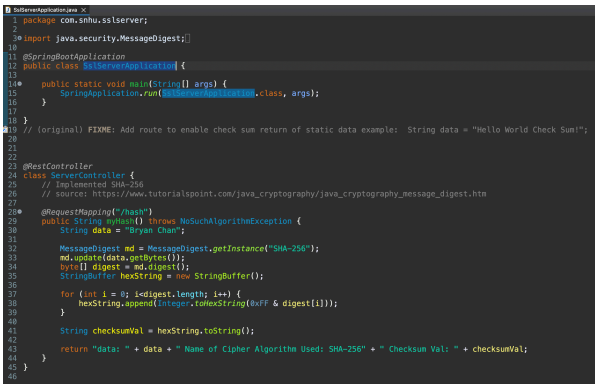
3. Deploy Cipher

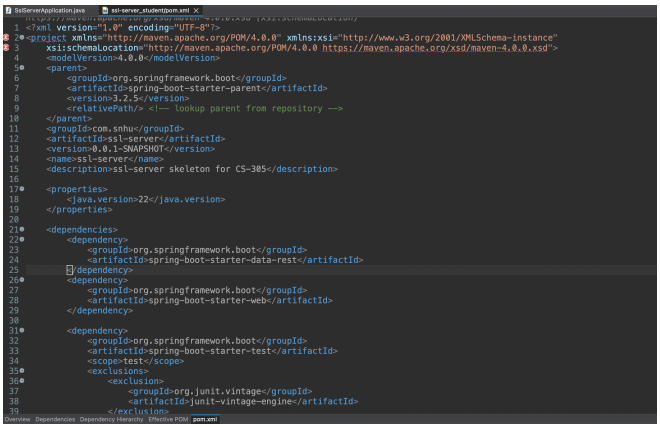
```java
package com.snhu.sslserver;

import java.security.MessageDigest;

@SpringBootApplication
public class SslServerApplication {

    public static void main(String[] args) {
        SpringApplication.run(SslServerApplication.class, args);
    }

}
// (original) FIXME: Add route to enable check sum return of static data example:  String data = "Hello World Check Sum!";



@RestController
class ServerController {
    // Implemented SHA-256
    // source: https://www.tutorialspoint.com/java_cryptography/java_cryptography_message_digest.htm

    @RequestMapping("/hash")
    public String myHash() throws NoSuchAlgorithmException {
        String data = "Bryan Chan";

        MessageDigest md = MessageDigest.getInstance("SHA-256");
        md.update(data.getBytes());
        byte[] digest = md.digest();
        StringBuffer hexString = new StringBuffer();

        for (int i = 0; i<digest.length; i++) {
            hexString.append(Integer.toHexString(0xFF & digest[i]));
        }

        String checksumVal = hexString.toString();

        return "data: " + data + " Name of Cipher Algorithm Used: SHA-256" + " Checksum Val: " + checksumVal;
    }
}

```

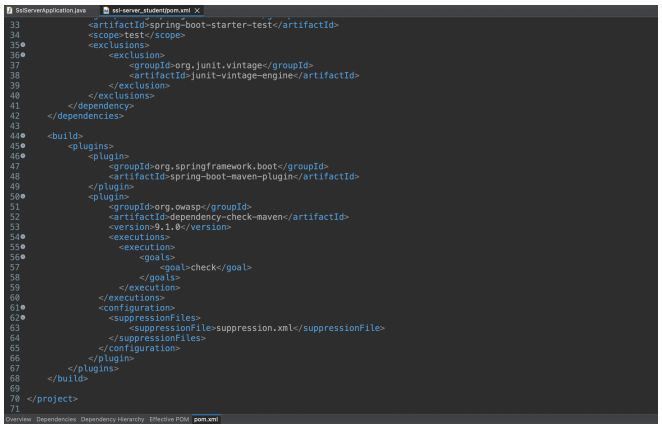
4. Secure Communications

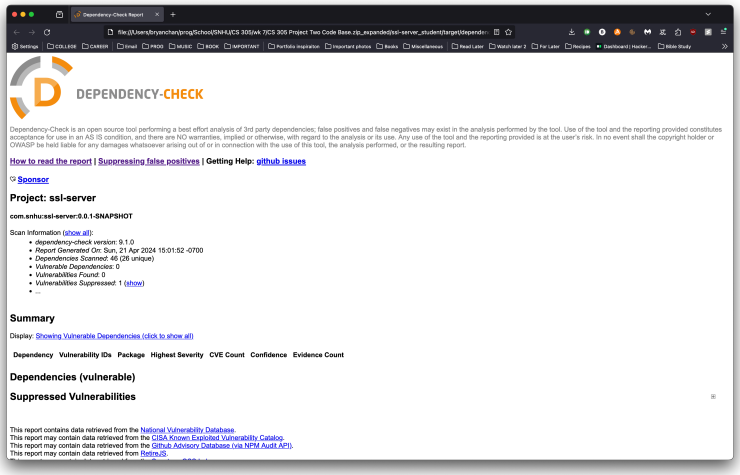


5. Secondary Testing

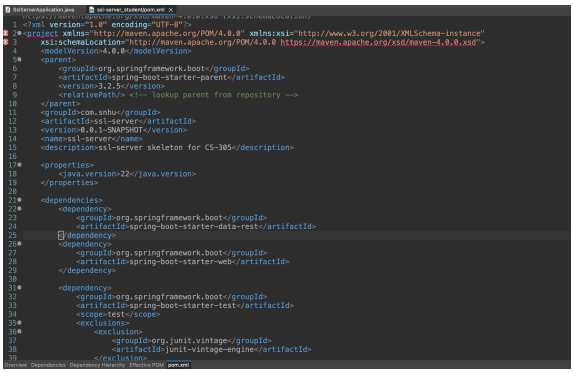


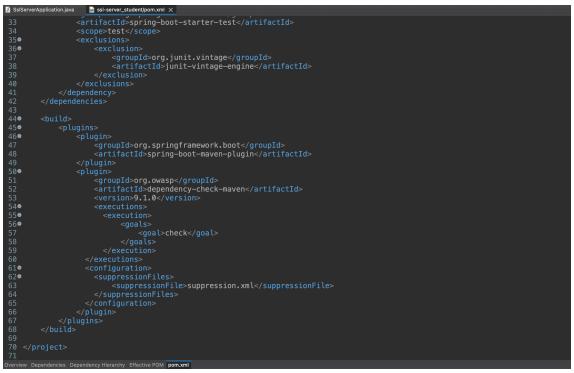


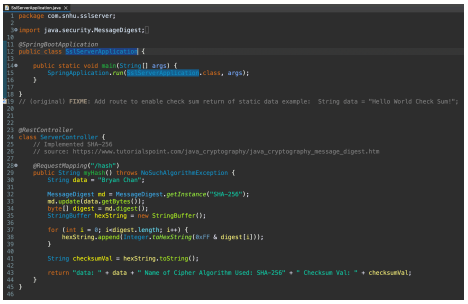




6. Functional Testing







7. Summary

The codebase for Project Two was refactored to incorporate the SHA-256 encryption algorithm. This fulfills the “cryptography” section of the Vulnerability Assessment Process Flow Diagram from SNHU (SNHU, 2024). The SHA-256 algorithm encrypts the input data from the codebase, in order to protect its contents from attackers. The algorithm does it in a manner that results in little-to-no collisions.

8. Industry Standard Best Practices

Throughout the codebase for Project Two, numerous best practices for security were implemented. First, the codebase contained a cryptographic algorithm such as the SHA-256, which provided encryption. Next, the codebase underwent static testing so that vulnerabilities would be detected. After updating the SpringBoot dependency, the program resulted in only one false positive vulnerability, which was later suppressed. Last but not least, a self-signed certificate was added so that the program would have a form of authenticity. Authenticity provided additional security to the program in order to prevent attackers from falsifying their identity.

References

Encryption Consulting. (2023, September 18). What is SHA? What is SHA used for? https://www.encryptionconsulting.com/education-center/what-is-sha/

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

SNHU. (2024). Vulnerability Assessment Process Flow [Diagram]. https://learn.snhu.edu/content/enforced/1535897-CS-305-R4825-OL-TRAD-UG.24EW4/course_documents/CS%20305%20Vulnerability%20Assessment%20Process%20Flow%20Diagram.pdf?_&d2lSessionVal=ScQoRYLN9OTGv4T9RbCyYwie6&ou=1332052&ou=1535897

Thales Group. (2023, June 10). A brief history of encryption (and cryptography). https://www.thalesgroup.com/en/markets/digital-identity-and-security/magazine/brief-history-encryption