# Southern New Hampshire University

# 7-1 Project Two

Anthony Spedaliere

Dr. Lyon

CS-305-T1165 Software Security

October 10th, 2022

****

# 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** | **10/10/2022** | **Anthony Spedaliere** |  |

## Client



## Developer

Anthony Spedaliere

## Algorithm Cipher

SHA-256 with RSA-2048

Artemis Financial seeks to add a file verification step to their web application. Java provides a list of security standard algorithms to achieve a higher standard of security (Java Security Standard Algorithm Names, n.d.-b). From this list the SHA-256 cipher in combination with the asymmetric RSA algorithm is the best suited to meet the needs of Artemis Financial.

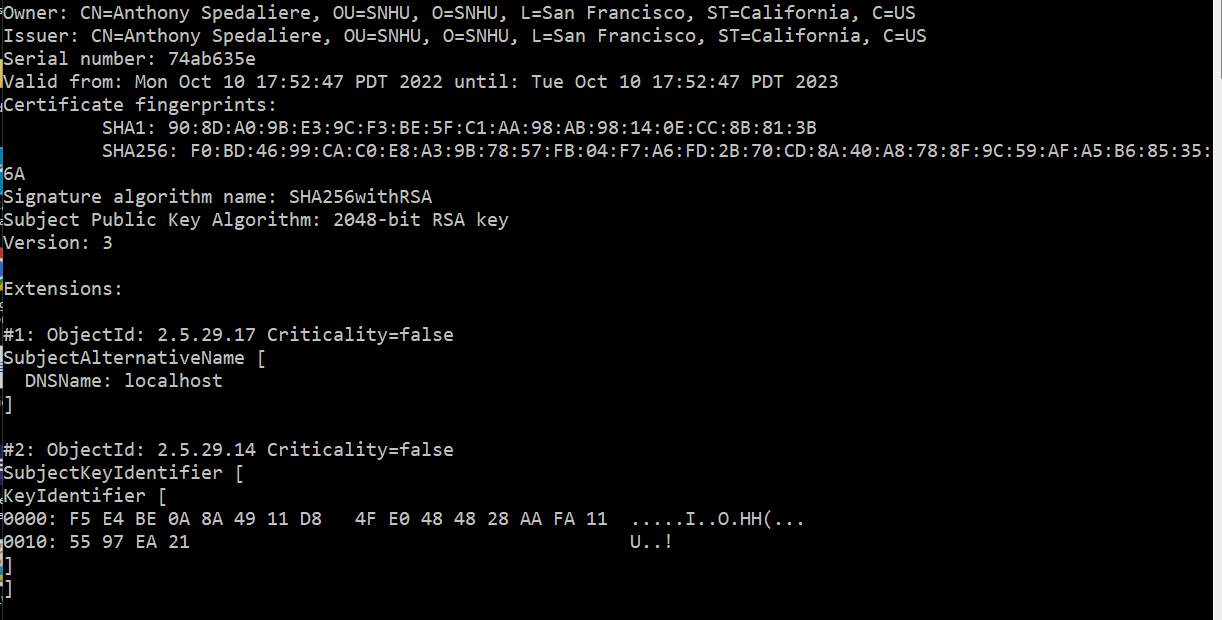
SHA-256 is a hashing algorithm that uses a 256-bit key to convert data of any length into a fixed-size length of 256-bits (Rhodes, 2022). It can also be used to encrypt files as opposed to just text, which is the application that is needed for Artemis Financial. SHA-256 is considered the industry standard and has a high collision resistance (Thakkar, 2022). The SHA cipher has various bit key length options such as 224, 256, 384, 512, 512/224, and 512/256 (Encryption, 2022). The recommended bit key length is 256.

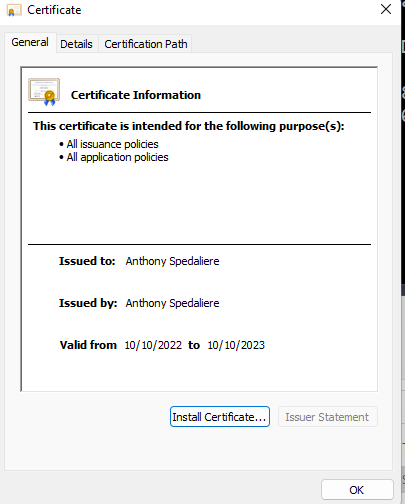
The RSA algorithm is an asymmetric algorithm and is ideal when there are two parties who need to verify trust and to ensure the data that is being exchanged can also be trusted. RSA works by using public and private keys (GeeksforGeeks, 2022c). According to GeeksforGeeks, “The Public Key of the receiver is used to encrypt the plaintext by the sender while the Private Key of the receiver is used to decrypt the encrypted message and hence can be decrypted only by the intended receiver. (2022c)” If even one character of the data is changed it can be detected. The current standard key-bit length is 1024-bits or 2048-bits (GeeksforGeeks, 2022c). The recommended key length for Artemis Financial is 2048-bits.

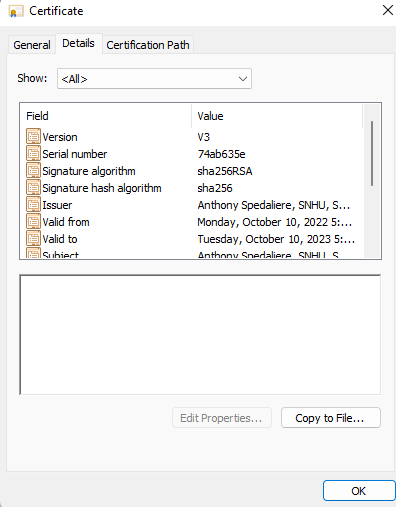
History has proven that over time computers have become faster and more efficient. As computers advance, the key lengths of algorithms will also need to be increased. That is why it is imperative that software engineers remain vigilant and provide secure software that contains standards above and beyond what is expected. By doing this we will remain one step ahead of those who wish to compromise the security of our digital spaces.

## Certificate Generation

## Certificate File Generation

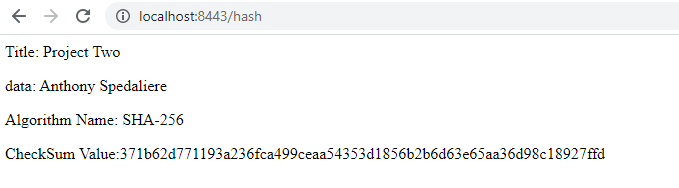






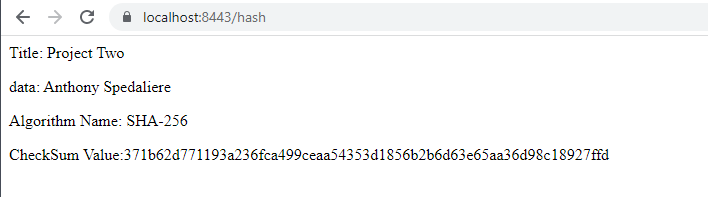
## Deploy Cipher

Checksum verification.



## Secure Communications

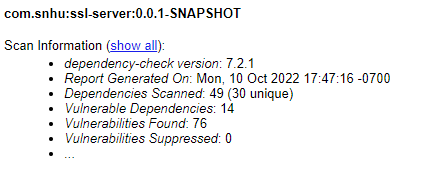
Secure webpage through a public web interface.



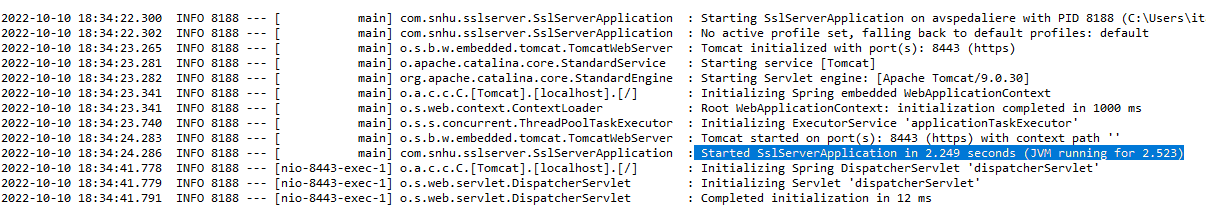
## Secondary Testing

Refactored code executed without errors and the dependency-check report.

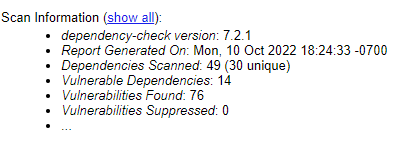
**Dependency Report Before Refactoring:**



**Executing Refactored Application:**



**Dependency Report After Refactoring:**



## Functional Testing

The current version of the program does not require input validation, because the data is hard coded. Below is line 34 taken from the SslServerApplication.java file showing the hard coded data.



However, the final program will require input validation since it is a requirement that the program accept user files from a user and verify them. The first step in this process would be **user verification** followed by **input validation**. User verification can be achieved through proper username and password followed by a two-step verification such as texting or emailing a short code. Additionally, the implementation of a checksum will help to verify that the file sent by the authenticated entity remains unchanged while enroute. If a single character were to change in the file, then the generated checksum would be different (Fisher, 2022). Line 35 of the refactored code shows the variable ultimately used to return the checksum (see below screenshot).



The checksum value is also a form of **cryptography** since it uses encryption/decryption of the original user data.

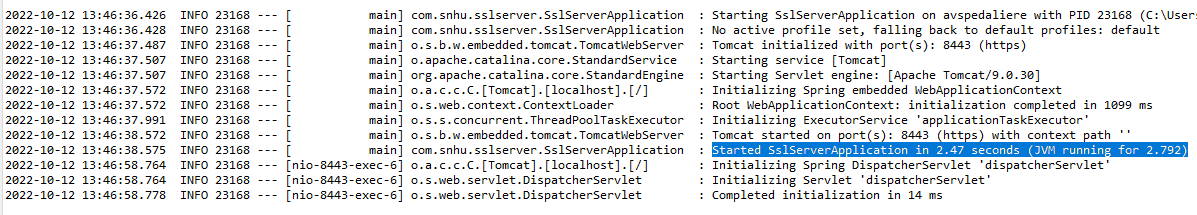
Other Input validation techniques are regular expressions, parameterized input values, min/max value range checks, and type conversions (Input Validation - OWASP Cheat Sheet Series, n.d.).

Since the user input will be files, **upload verification** will be essential. This can be achieved by checking file extension types, ensuring the uploaded file is not too large, and validating any zipped files before they are unzipped (Input Validation - OWASP Cheat Sheet Series, n.d.). Additionally, when storing user files, it is important to rename the user files to prevent any future direct file access by malicious entities (Input Validation - OWASP Cheat Sheet Series, n.d.).

The website was originally standard HTTP. This poses a threat to the website as it makes it vulnerable to hackers (Williams, 2022). Converting the site to HTTPS encrypts the data as it travels from the **client**/user to the **server** and helps to ensure that the data integrity is not compromised by using **encryption**.

Lastly, ensuring maximum **code quality** and the code follows secure practices and patterns. The primary pattern to implement in this project is the model view controller framework or MVC. The MVC framework separates the application into three areas – the model, view and controller (GeeksforGeeks, 2022b). By funneling all the user input through the controller, it allows us to implement our security checks before it ever reaches the backend model.

**Executing Refactored Application:**



## Summary

The areas I addressed during the code refactoring were input validation, upload verification, cryptography, client/server and code quality.

The two ways I refactored the code to make it more secure were by implementing a checksum and converting from HTTP to HTTPS. The checksum uses encryption to allow one party to send data to another party and ensure that the data that is received is the same as the data that was sent – whether by accident or intentional. My checksum is implemented using SHA-256 hashing algorithm. The second way the code was refactored was by implementing TLS by converting from HTTP to HTTPS. By using asymmetric public key infrastructure, the site can create trust between two parties and files exchanged by the parties will be encrypted to ensure that they are not intercepted and altered en route.

These extra layers of security will ensure that Artemis Financial can safely verify files from trusted parties and store them on their server.

## Industry Standard Best Practices

The most current industry best standard for web applications encryption is to implement Transport Layer Security (TLS) protocols (Security Developer’s Guide, n.d.). It is the most widely used protocol for implementing cryptography on the internet (Security Developer’s Guide, n.d.). I was able to successfully implement this protocol as illustrated in part 4 of this report. TLS protocol protects the web application in various ways such as through secret-key cryptography, public-key cryptography, public-key certificates, cryptographic hash functions, message authentication codes, and digital signatures (Security Developer’s Guide, n.d.). The value of applying industry best standards to the Artemis Financial web application is vital. Industry best practices are best practices because they are the culmination of many different ideas and exercises. The best ideas are selected and the ones with the best results are implemented by the industry. That is why it is important to pay attention to the current trends so that when the industry best practices evolve and change the web applications can implement these changes.

**References**

*Java Security Standard Algorithm Names*. (n.d.-b). Retrieved October 10, 2022, from https://docs.oracle.com/javase/9/docs/specs/security/standard-names.html

Rhodes, D. (2022, April 29). *SHA-256 Cryptographic Hash Algorithm*. Komodo Academy | En. Retrieved October 10, 2022, from https://komodoplatform.com/en/academy/sha-256-algorithm/

Thakkar, M. (2022, April 18). *SHA 256 Algorithm Explained by a Cyber Security Consultant*. InfoSec Insights. Retrieved September 26, 2022, from https://sectigostore.com/blog/sha-256-algorithm-explained-by-a-cyber-security-consultant/

Encryption. (2022, April 22). *SHA 256 Algorithm Explained by a Cyber Security Consultant*. Infosec Insights. Retrieved October 11, 2022, from https://sectigostore.com/blog/sha-256-algorithm-explained-by-a-cyber-security-consultant/

GeeksforGeeks. (2022c, September 26). *RSA Algorithm in Cryptography*. Retrieved October 11, 2022, from https://www.geeksforgeeks.org/rsa-algorithm-cryptography/

*Input Validation - OWASP Cheat Sheet Series*. (n.d.). Retrieved October 12, 2022, from https://cheatsheetseries.owasp.org/cheatsheets/Input\_Validation\_Cheat\_Sheet.html

Fisher, T. (2022, February 11). *What Is a Checksum? See a Definition, Examples, and More*. Lifewire. Retrieved October 12, 2022, from https://www.lifewire.com/what-does-checksum-mean-2625825

Williams, L. (2022, September 23). *HTTP vs HTTPS – Difference Between Them*. Guru99. Retrieved October 12, 2022, from https://www.guru99.com/difference-http-vs-https.html

GeeksforGeeks. (2022b, March 22). *MVC Framework Introduction*. Retrieved October 12, 2022, from https://www.geeksforgeeks.org/mvc-framework-introduction/

*Security Developer’s Guide*. (n.d.). Oracle Help Center. Retrieved October 12, 2022, from https://docs.oracle.com/en/java/javase/15/security/transport-layer-security-tls-protocol-overview.html