****

# 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 6](#_Toc102040760)

[4. Secure Communications 6](#_Toc102040761)

[5. Secondary Testing 6](#_Toc102040762)

[6. Functional Testing 7](#_Toc102040763)

[7. Summary 8](#_Toc102040764)

[8. Industry Standard Best Practices 9](#_Toc102040765)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **10/10/2022** | **Anthony Vigil** |  |

## 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

Anthony Vigil

## Algorithm Cipher

As we will be optimizing security features for the given code, my recommendation would be to utilize the encryption algorithm cipher known as the Advanced Encryption Standard, AES. AES 256 will provide more security for communications. AES is a symmetric key cipher that allows the same key to be utilized for both the encryption and the decryption, which ultimately requires both the sender and the receiver of the data to have a copy of the key [2]. An advantage of using AES 256 is the speed due to symmetric key algorithms requiring less computational power.

Secure hash algorithms, SHAs, are designed to keep the user’s data secured. It is utilized by transforming the data using a hash function, which can be characterized as bitwise operations, compression functions, or modular additions [3]. The hash function then produces a string that will appear to be different from the original text; This will ultimately make it very difficult to change back to the original. Reading the bit level lets us know the security of the cipher, such as AES 128, 192, 256 bit levels. Ultimately, it means the higher the bit level, the more secure the encryption resulting in AES 256 being the most secure.

As mentioned above, AES is a symmetric key cipher which allows for the same key to be used to send and retrieve messages. Symmetric encryptions use a single key, while asymmetric encryption uses public-private keys but will take more time to complete when compared to symmetric encryption [4]. Big data is best transmitted through symmetric encryptions, while small data is best transmitted by asymmetric encryption. To ensure security, the keys used for encryption will need to be random to prevent hackers from being able to figure out the key.

Today, the commonly used encryption algorithms are Triple Data Encryption Standard (DES), Advanced Encryption Standard (AES), and RSA security, Blowfish, and Twofish [1]. As many are being tested for their invulnerabilities, the future of encryption is looking at the impacts of quantum cryptography for the future of encryption.

## Certificate Generation

Insert a screenshot below of the CER file.

![Text

Description automatically generated]()

![Graphical user interface, text, application, email

Description automatically generated]()

## Deploy Cipher

Insert a screenshot below of the checksum verification.

![Graphical user interface, application

Description automatically generated]()

## Secure Communications

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

Graphical user interface, application, Teams

Description automatically generated

I imported the certification and also installed it to the user, which should enable securing the site as I have the key. After installing and importing, it still did not work to secure the webpage.

## Secondary Testing

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

Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

## Functional Testing

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

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

## Summary

Referring to the Vulnerability Assessment Process Flow Diagram, security protocols were implemented throughout the process.

APIs: API was implemented when the security verification of HTTPS on the web browser was confirmed. This allowed us to secure communication of both the sender and receiver.

Cryptography: Cryptography is implemented when working with the hash function after the encryption algorithm cipher. The hash function was verified using the check sum.

Client/Server: During the testing process for the application, the client and server used remained the same. In sending the message during communication, the receiver was able to receive the message as well; This implementation allowed us to verify that the information is transferred.

Code Error: To analyze the code, an exception that helped identify errors was the NoSuchAlgorithm exception that is included in the myHash method. This allowed for the error to be displayed in the bottom of eclipse if any information is not presented.

Code Quality: Readability and functionality is implemented in this section. The code is refactored to include the new functions but also allows for other employees or teammates to read the code with ease.

In this project, we added security to the application that enabled using the HTTPS on the webpage to have secured communication. It was done by making a self-signed certificate from the command line using the keytool. The pom.xml file from the code base was refactored and then put through a dependency check to find new or existing invulnerabilities.

The first task given was to create the self-signed certificate, this was done by utilizing keytool.exe, in which the details for the certification were written and then a password was created for it. Then, move the file to another location as a certificate file, but also save the jks file in a known location.

After saving, the next step was to ensure the hash function works with a check sum verification. The issue I experienced was an exception involving Java, but after some research the issue came out to be the location of the jks file was not in the same directory of the project. After moving it to the same directory, the check sum verification was complete. The localhost: 8443/hash showed the string data and name as it was shown in the code. This allows for the user’s data to be secured by being sent and retrieved from other users.

The last step was to determine If the vulnerabilities were resolved. This was established by analyzing the code for errors and making sure the application works properly. Establishing this security gives ensured prevention of hacks and allows for multiple data to be protected.

## Industry Standard Best Practices

Industry standard best practices include keeping the application updated and the new vulnerabilities resolved. Hackers usually target out of date systems which allows user data to get stolen or breach into systems. Keeping the entire system updated ensures no potential invulnerabilities are taken advantage of by outsiders. By using the standard best practices, we are preventing new vulnerabilities and isolating them as they arise.

References

“5 Common Encryption Algorithms and the Unbreakables of the Future.” Arcserve, https://www.arcserve.com/blog/5-common-encryption-algorithms-and-unbreakables-future#:~:text=2.-,AES,for%20heavy%2Dduty%20encryption%20purposes.

N-able. “Advanced Encryption Standard: Understanding AES 256 - N-Able.” N, 22 Aug. 2022, https://www.n-able.com/blog/aes-256-encryption-algorithm.

“Secure Hash Algorithms.” Brilliant Math &amp; Science Wiki, https://brilliant.org/wiki/secure-hashing-algorithms/.

“Symmetric vs. Asymmetric Encryption - What Are Differences?” SSL2BUY Wiki - Get Solution for SSL Certificate Queries, 14 June 2021, https://www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences#:~:text=Difference%20Between%20Symmetric%20and%20Asymmetric,and%20decrypt%20messages%20when%20communicating.