### Southern New Hampshire University

### Project Two: Practices for Security Report

Joaquin Esguerra Jr

### Dr. Lyon

### CS 305 Software Security 22EW3

February 18, 2022

****

# 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** | **February 18, 2022** | **Joaquin Esguerra Jr** |  |

## 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

Joaquin Esguerra Jr

## 1. Algorithm Cipher

Determine an appropriate encryption algorithm cipher to deploy given the security vulnerabilities, justifying your reasoning. Be sure to address the following:

* Provide a brief, high-level overview of the encryption algorithm cipher.
* Discuss the hash functions and bit levels of the cipher.
* Explain the use of random numbers, symmetric vs non-symmetric keys, and so on.
* Describe the history and current state of encryption algorithms.

Based on the security vulnerabilities presented, the algorithm cipher I would recommend is the Advanced Encryption Standard (AES) bit 256. In 2001, the Data Encryption Standard (DES) proved to lack the necessary features needed for security and was replaced by AES which has been utilized by the U.S. Government since that time. Utilized in both Public and private sectors, AES is known to be the standard of encrypting data.

When looking at hash functions, the most widely used is SHA, or the Secure Hashing Algorithm. To highten security, AES can be partnered with SHA which converts plaintext to ciphertext which is difficult to decipher. Gauging a specifc bit level, in terms of security, translates to the amount of operations a potential threat would nbeed to perform to crack cipher. Advanced Encryption Standard offers multiple bit levels ranging from 128 to 256, with 192 bit being the mid level option. Utilizing AES-256 will provide the highest level of security needed for this scenireio.

In comparing encryption ciphers, one thing to note is some diffrences between symmetric encryption and asymmetric encryption. Symmetic encryption utilizes a single key which is shared amongst users to receive data, asymmetric uses both a public and private key for encytping and decrypting communication. AES, being a symmetric cipher, encrypts data block consistently through the use of long key lengths which hightens security. A important point to notes is how keys should be random, as keys easily memorized are the ones that present the potential for threats.

Modern cryptography, or computer-based encryption is what keeps our personal data secure. It can be described as an egg cracked open into a bowl and then whisked. The whisking of an egg is what an encryption does when it scrambles our personal data to prevent potential threats from misuse of the information. Cryptography began by Ancient Spartans in 600 BC. Spartans used a device known as a scytale to secretly communicate. The scytale was made up of a leather strap and rod. Only with the correct rod tould the message be depicted. Later on the Romans began encryption and cyphers to communicate. During 60 BC Juilius Caesar inveted a cypher that substituted letters which during that period proved to be effective. In America, around 1917 a man by the name of Edward Hebern invented a machine known as the “Hebern rotor machine” which used keys embedded in a rotaing disc. The machine encrypted messages every time a single character was type. Year after year more devices would be introduced into society. All proved effective for their use during the periods they exsited. It wasn’t until the early 1970s when the first modern computer-based encryption came to light. Known as the Data Encryption Standard (DES), the U.S. Government quickly adopted DES and it was know as the standard until 1997 when its securities failed to protect against potential threats. Today AES, or Advanced Encryption Standard has replaced DES and is available worldwide. Known for its secure infastructure, AES is approved and used by the U.S. government to protect all classified information.

## 2. Certificate Generation

Generate appropriate self-signed certificates using the Java Keytool, which is used through the command line.

* To demonstrate that the keys were effectively generated, export your certificates (CER file) and submit a screenshot of the CER file below.

Text

Description automatically generated

## 3. Deploy Cipher

Refactor the code and use security libraries to deploy and implement the encryption algorithm cipher to the software application. Verify this additional functionality with a checksum.

* Insert a screenshot below of the checksum verification. The screenshot must show your name and a unique data string that has been created.

Text

Description automatically generated

## 4. Secure Communications

Refactor the code to convert HTTP to the HTTPS protocol. Compile and run the refactored code to verify secure communication by typing **https://localhost:8443/hash** in a new browser window to demonstrate that the secure communication works successfully.

* Insert a screenshot below of the web browser that shows a secure webpage. (I executed the extra TIPS from your announcement on 2/16, but was unable to connect to the localhost. I even created a additional CA based on the tips and still no luck.)

Text

Description automatically generated

## 5. Secondary Testing

Complete a secondary static testing of the refactored code using the dependency check tool to ensure code complies with software security enhancements. You only need to focus on the code you have added as part of the refactoring. Complete the dependency check and review the output to ensure you did not introduce additional security vulnerabilities.

* Include the following below:
  + A screenshot of the refactored code executed without errors
  + A screenshot of the dependency check report

Text

Description automatically generated

## Graphical user interface, text Description automatically generated

## 6. Functional Testing

Identify syntactical, logical, and security vulnerabilities for the software application by manually reviewing code.

* Complete this functional testing and include a screenshot below of the refactored code executed without errors.

Text

Description automatically generated

Text

Description automatically generated

## 

## 7. Summary

Discuss how the code has been refactored and how it complies with security testing protocols. Be sure to address the following:

* Refer to the Vulnerability Assessment Process Flow Diagram and highlight the areas of security that you addressed by refactoring the code.
* Discuss your process for adding layers of security to the software application and the value that security adds to the company’s overall wellbeing.
* Point out best practices for maintaining the current security of the software application to your customer.

1. Code quality – Good (High Quality) code that maintained simplicity, is readable, and can be easily contributed to, along with ease it ability to analyze.
2. Code errors – Addressed by understanding excpetions in a given class through the retuned code. Primarily the NoSuchAlgorithm excpetion within the myHash() method.
3. Client/Server – The model was primarily the same through testing, but addressed tasks or workloads that would be sent between servers and clients.
4. Cryptography – Addressed through the encryption algorithm cipher, hash functions, along with the checksum verification used within the code base
5. Application Program Interface – Called API specificstion, they were addressed by implementing HTTPS and providing an interface viewable by the user

The primary added security for the application was a Certificate Authority, that was self-signed allowing the HTTPS to be utilized. Refractoring the pom.xml file also ensure greater security by clearing up vulnerabilities which were discovered during dependency checks.

The intial step was to aquire a self-signed certificate set up in a way to easily utilize HTTPS once the full application was up and running. The certificate would be the first layer of security for the organization in knowing the web page would be secure and that clients/users could feel confident in browsing with us on the other end.

Once initialized, the hash function provided the next level of security. Verfication using the checksum ensured the organization that all data was hashed correctly and all users could be confident that data would not be easily accessed by potential threats.

Finally, suppression of known vulnerabilities was key in covering the organization across the board as it moved forward will application processes and ensured systems were updated and functioning as expected.

A best practice to live by is the demonstration of maintaining security patches in systems and software through an application. Having updated current systems blocks potential threats which if out dated could be exsposed. As an additional precaution, enabling a least privilege would also prove beneficial as interal threats can be a problem in todays society and this would allow for access to be restricted on a as needed bases within the organizations group of users.

References

*A brief history of encryption*. Thales Group. (2021, October 1). Retrieved from https://www.thalesgroup.com/en/markets/digital-identity-and-security/magazine/brief-history-encryption

*Comparison of various encryption algorithms for securing data*. (n.d.). Retrieved from http://eprints.rclis.org/33889/1/comparison%20report%20.pdf

Dworkin, M. J., Barker, E. B., Nechvatal, J. R., Foti, J., Bassham, L. E., Roback, E., & Jr., J. F. D. (2021, March 1). *Advanced encryption standard (AES)*. NIST. Retrieved from https://www.nist.gov/publications/advanced-encryption-standard-aes

Manico, J., & Detlefsen, A. (n.d.). *Iron-clad Java*. O'Reilly Online Learning. Retrieved from https://learning.oreilly.com/library/view/iron-clad-java/9780071835886/ch06.html

Person. (2022, February 2). *What is least privilege & why do you need it?* BeyondTrust. Retrieved from https://www.beyondtrust.com/blog/entry/what-is-least-privilege

*What is secure hashing algorithm (SHA): SHA1 vs SHA2*. https.in Blog. (2021, December 17). Retrieved from https://www.https.in/blog/secure-hashing-algorithm/