****

# 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** | **[04/22/2024]** | **[Junyu Liang]** |  |

## 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

[Junyu Liang]

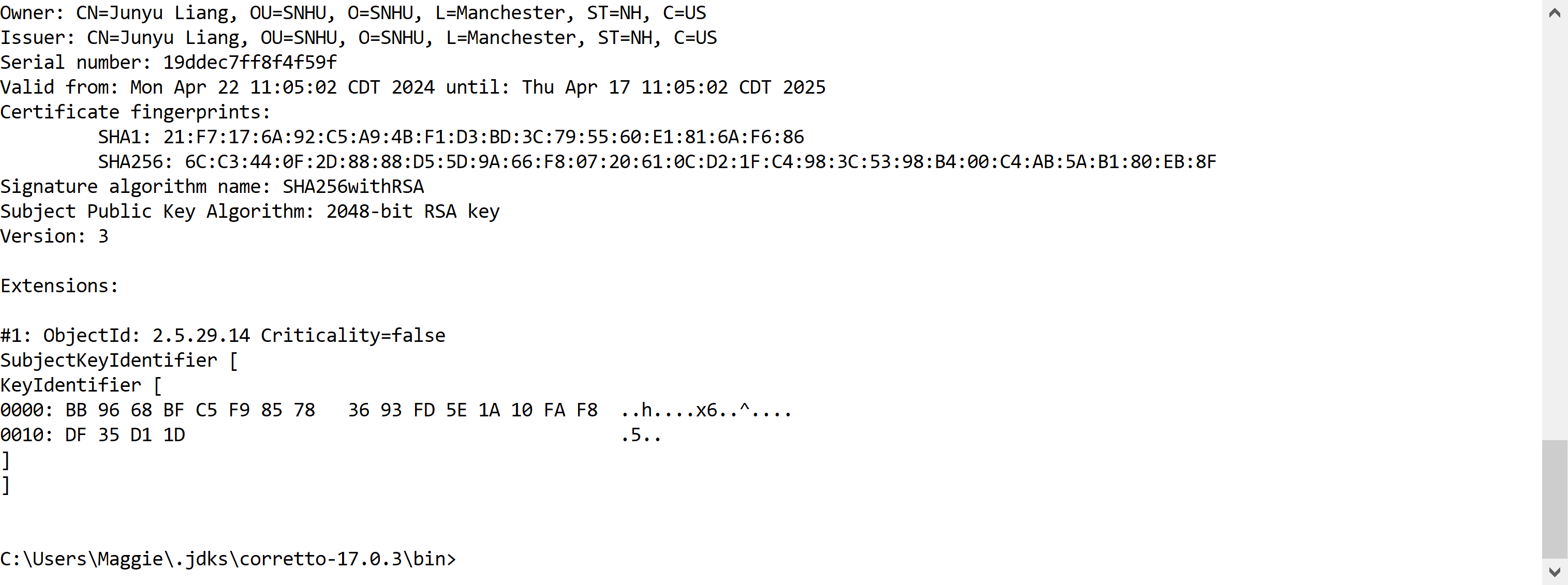
## Algorithm Cipher

When developing this system, I chose SHA-256 encryption algorithm. This is because SHA-256 is a 256-bit encryption algorithm. The 256-bit encryption algorithm provides collision prevention and is difficult to crack in a human lifetime, even for supercomputers (Saulius Griškėnas. 2023.). Cracking 256-bit encryption takes 3.31\*10^56 years. Hash functions are used for encryption algorithms, including the one I use. These functions convert readable data to unreadable data, and the process is one-way. Once the data is hashed, you can't really retrieve the data without accessing the key. The hash function in this password takes any input and outputs a long string of 64 bits, basically random text. There are two main types of keys in cryptography. The first is symmetric. Symmetric cryptography uses a single key to encrypt and decrypt information. This key must be kept in a secure location because if it is lost, any data encrypted with this key cannot be recovered. Then, there is asymmetric key encryption that works in different ways. Asymmetric keys encrypt data using a key, which is usually public facing(Kate Brush). This also utilizes random number generation to ensure that keys do not overlap, or at least are highly unlikely to. In addition, the keys must be independent of each other. This means that the person who has access to the public key should not give him any way to get the private key. Encryption has a long history. For a long time, people have sought content that hides their information from prying eyes. While the method of hiding the content of a message is very interesting, we are talking about modern cryptography here. This stems from the Caesar Box and Viginere cipher, followed by the Enigma Cipher machine of World War II. As early as 1971, IBM created the "Lucifer" encryption technology for civilian use (KONACR **.** OCTOBER 22, 2019). Asymmetric cryptography was also first developed in the 1970s, and since then it has exploded in both complexity and private and public use. For anyone who cares about security, the days of 56-bit encryption are long gone. Modern algorithms with 256 bits as the current standard are much more robust, with options up to 2048 bits.

## Certificate Generation

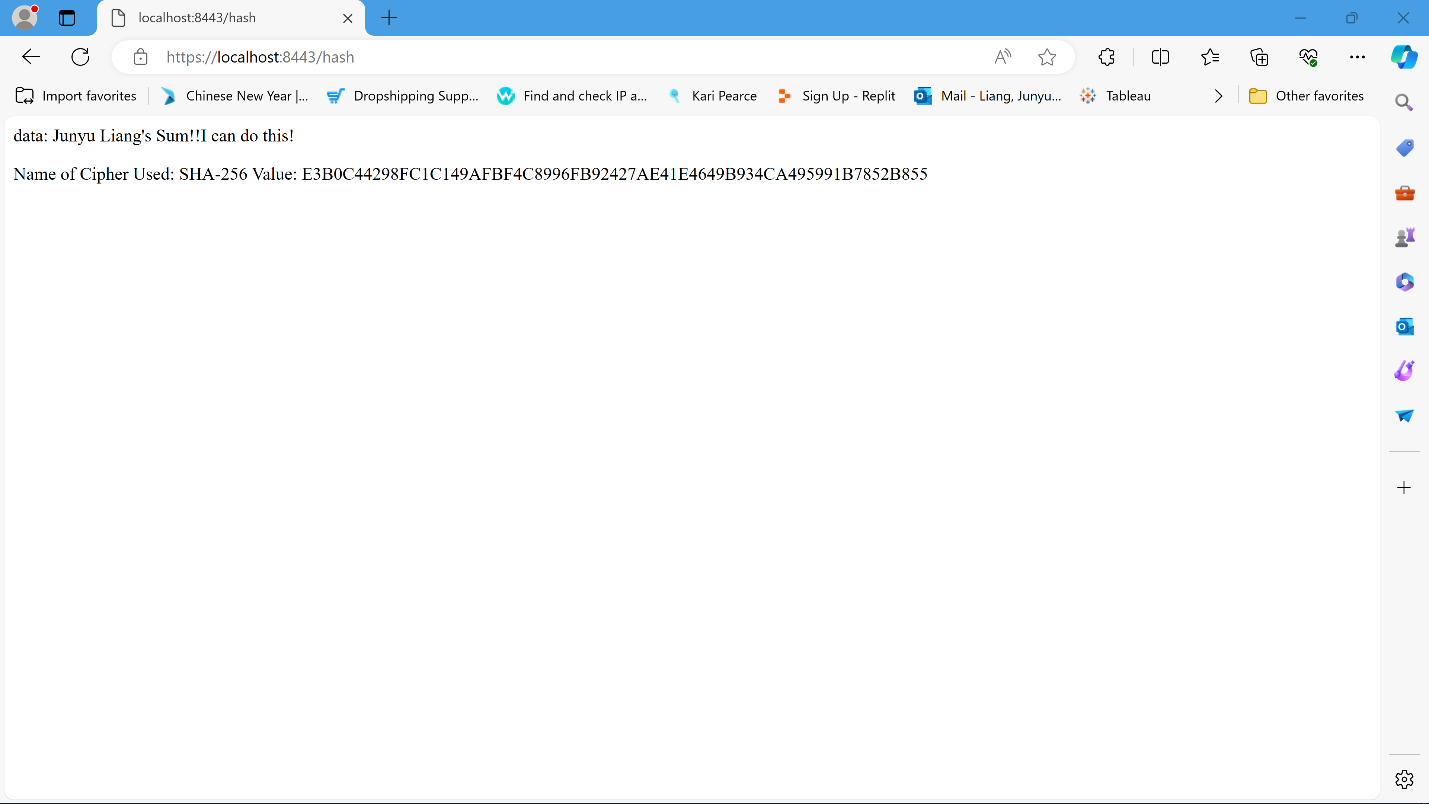
## Create self-signed certificates using Java Keytool via the command line.

Insert a screenshot below of the CER file.



## Deploy Cipher

Insert a screenshot below of the checksum verification.



## Secure Communications

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

## Secondary Testing

## Before refactering

## 

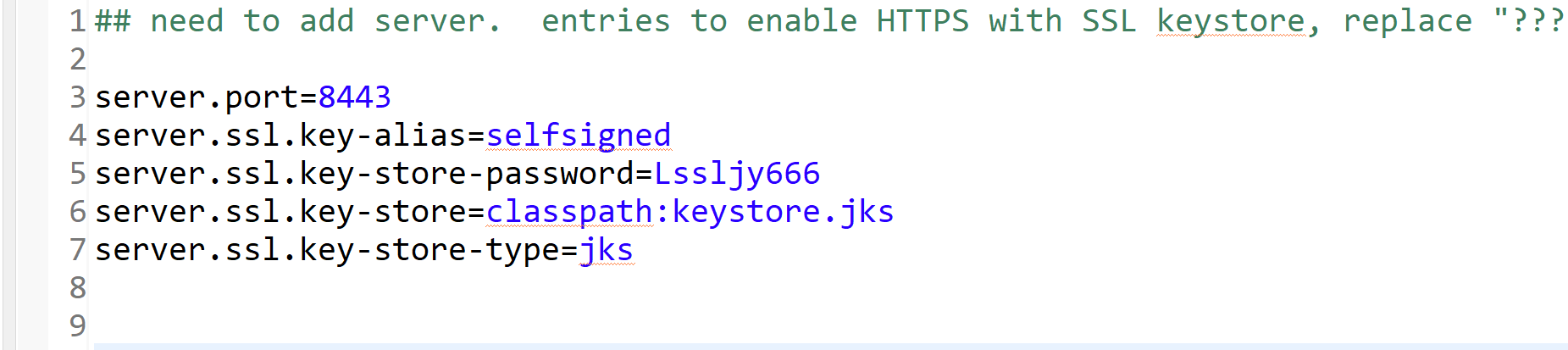
## After refactoring

## 

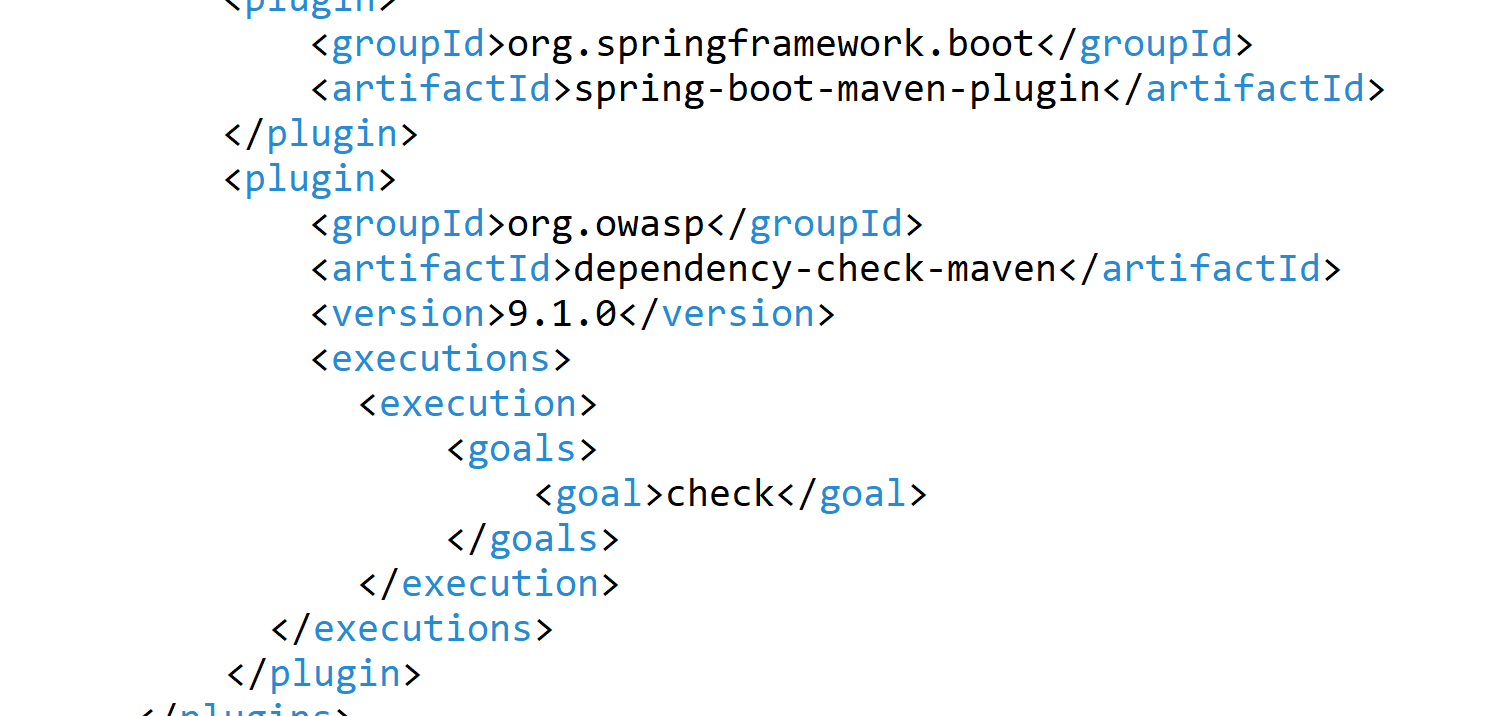
## 5

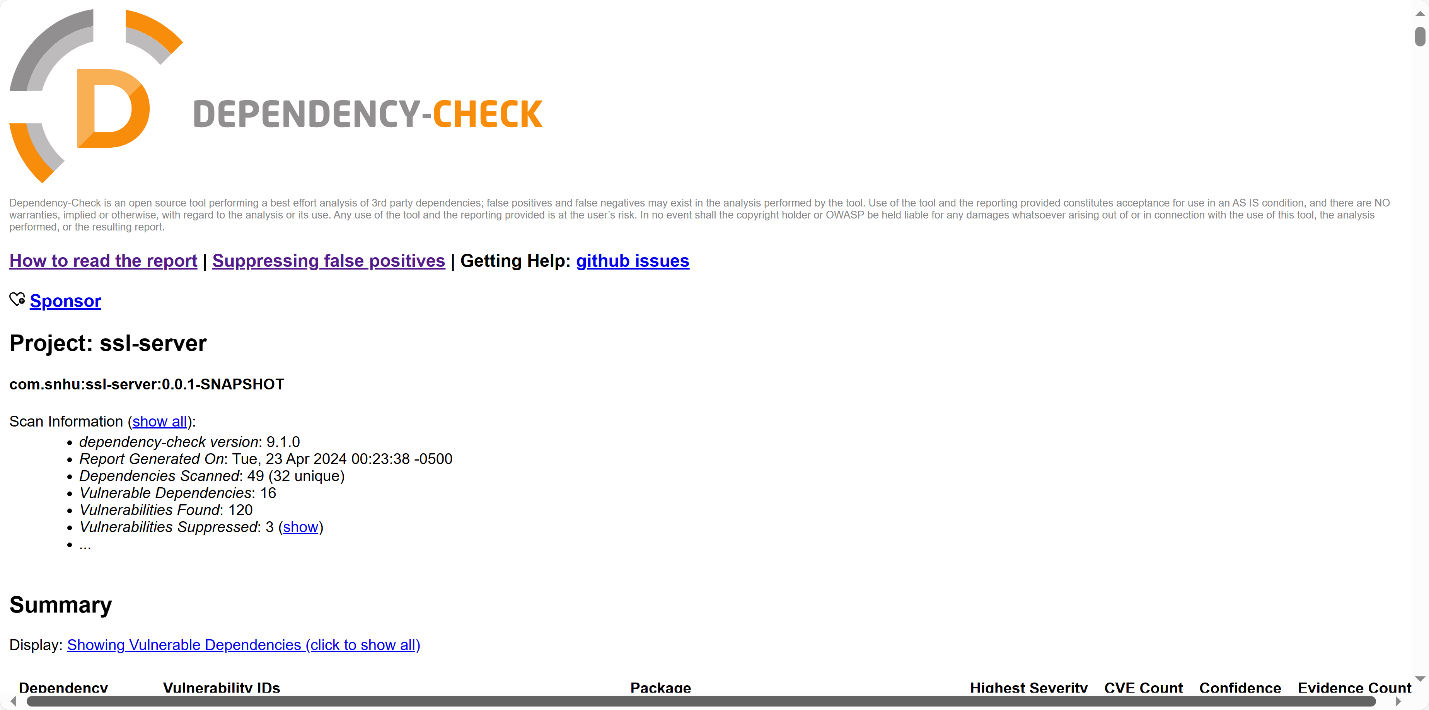
Insert screenshots below of the refactored code executed without errors and the dependency-check report.

The code has been executed with no errors.



Updated the lastest version of in the pom.xml file.





## 6.Functional Testing

## Code is manually examined to identify syntax, logic, and security vulnerabilities in software applications

## 

## 7.Summary

In this project, I focused on cryptography, server/client security, and code quality. I improve code quality by combining secure coding practices and implementing a 256-bit hashing algorithm to protect sensitive data such as names or financial information for Artemis financial. For client/server security, I integrated SSL certificates and enforced the use of TLS connections to secure communications. This ensures that both the connection and the data are encrypted. In addition, I mandated HTTPS to prevent insecure connections and added SHA-256 encryption for static data. All code is thoroughly reviewed to identify and correct vulnerabilities. Regular dependency checks, secure coding practices, input validation, and API design are also part of maintaining software security(Kate Brush . March 2024). These measures add a layer of security, ensure data protection and build user trust for Artemis Financial.

## 8.Industry Standard Best Practices

To maintain the security of the software, I apply industry best practices, such as using a trusted certificate authority for security certificate generation. I often use OWASP Dependency Check to monitor third-party components for vulnerabilities and keep them updated. In terms of encryption, I use strong algorithms like AES to protect the data. I also conducted regular security assessments and code reviews to identify and fix potential vulnerabilities.

Applying these practices is critical to the health of the company. It prevents data breaches, maintains regulatory compliance, and builds customer trust. Investing in security reduces financial and legal risk and contributes to a company's success and reputation.

Reference:

*What is the SHA-256 algorithm, and how does it work?* Saulius Griškėnas. Jun 20, 2023. Website: https://nordvpn.com/blog/sha-256/

# *The History and Mathematics of Codes and Code Breaking.* KONACR . OCTOBER 22, 2019. Website:

https://derekbruff.org/blogs/fywscrypto/tag/lucifer/

# *asymmetric cryptography.* Kate Brush . March 2024

# Website :https://www.techtarget.com/searchsecurity/definition/asymmetric-cryptography