****

# 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** | **04/16/2023** | **Ronald W. Sudol III** | **Rip and Tear!** |

## Client:



## Developer:

Ronald W. Sudol III

## Algorithm Cipher

Given that Artemis Financial will have sensitive financial data in their long-term archive files, the implementation of file encryption is essential to ensuring the security of their data. There are a variety of standards and protocols available for secure encryption of data. The cipher algorithm I would recommend for Artemis Financial is the Advance Encryption Standard (AES) with a 128-bit key. “Currently, AES is one of the best encryption protocols available, as it flawlessly combines speed and security, letting us enjoy our daily online activities without any disruption”(Rimkiene, 2020). Data encrypted with an AES 128 bit key also takes up far less space than the same data encrypted with other ciphers, line RSA for instance. For Artemis Financial, this means they can efficiently and securely store sensitive data while minimizing the amount of space that encrypted data will take up on a database server. Attempting to brute-force a 128-bit key would take far longer than a human lifetime, so the level of security provided is more than sufficient.

AES creates a symmetric key generated by implementing a random number. This means there is only one key with which the data is encrypted and decrypted and that the key is produced in a way that is random so that the key cannot be deduced. Other layers of security such as hashing the keys themselves and storing keysets in a separate database server can add further levels of security by ensuring the keys themselves remain secure. “A common saying in cryptography is that cryptography turns the problem of protecting a lot of information—your data—into the problem of protecting a little information: the keys”(Manico & Detlefsen,2015).

Encryption algorithms have existed since ancient times in the form of simple substitution algorithms like the Caesar cipher. With advances in technology, the complex and evolving landscape of cryptography and data security requires more advanced methods for organizations to be able to ensure the security of their data. The AES cipher is trusted by the US government for the protection of classified information which speaks highly of its current state, but it is important to know that new threats to security arise every day. With this in mind, it is important that Artemis Financial is vigilant in its handling of not only sensitive data but also its handling of key sets for encrypting and decrypting that data.

## Certificate Generation

Text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

## Deploy Cipher

Text

Description automatically generated

## Secure Communications

A screenshot of a computer

Description automatically generated

## Secondary Testing

Text

Description automatically generated

Graphical user interface, text, application, Teams

Description automatically generated

## Functional Testing

Graphical user interface, text

Description automatically generated

## Summary

In refactoring this code, I implemented several steps each with adding an additional layer of security in mind. These layers of security work in conjunction to provide a more robustly secure line of defense against malicious actors.

First, I made a recommendation for end-to-end encryption utilizing AES algorithms. This helps to keep data secure as it is transported. Then I utilized Java Keytool in Eclipse to generate a selfsigned certificate. I installed this certificate in the local server’s root certificate store so that it would be recognized as safe. Next, I implemented code and a hash algorithm for the purpose of generating a Checksum value for a predetermined string value. If anything about this string was altered, the checksum would be altered, thus creating an additional layer of security. Then I altered the information in the applications.properties file to reflect the keystore file I previously created when generating the certificate. Utilizing this keystore and certificate, the web server is now configured to operate along HTTPS protcols. Finally, I turned to the Maven dependency check. I noticed the code was factored to use an outdated version of Maven so I first updated the .xml file to utilize version 8.2.1. This brought up a considerable number of new dependencies. By creating a suppression file and adding a configuration plugin to the .xml file, I was able to weed out false positives and focus on the most concerning vulnerabilities. In reviewing these refined vulnerabilities and comparing them to the vulnerabilities present at the beginning of the project, no new vulnerabilities were created as a result of my refactoring of the code.

Functional testing and refactoring were things that occurred iteratively throughout the entire execution of this project. The most noteworthy is that the bytesToHex() function was not predefined for the java.security.MessageDigest import. I solved this by defining the function myself. Further, MessageDigest required the import of NoSuchAlgroithmException and StandardCharsets to function. The string used to create the checksum is predefined and not generated from user input so there is no chance of injection attacks.

## Industry Standard Best Practices

In terms of industry standard best practices, the practice of reiterating through the development process is important. In this project, keeping security in mind throughout those iterations was essential. Updating the Maven version was the right call, in terms of industry practices, because utilizing the most up to date libraries, extensions, and frameworks is always recommended. Furthermore, suppressing false positives in the dependency check allows a DevSecOps developer to focus on the most certain and critical vulnerabilities first and foremost.

Industry standard best practices are valuable to a company’s overall wellbeing. These practices ensure that dependency vulnerabilities are up to date, which allows the company to protect itself from the most current and common vulnerabilities. They aid the company in developing code that is secure and does not leave itself open to injection attacks. Further, these practices allow the company to clearly identify itself and to verify the identity of entities with which it interacts.

1. **Citations**

Rimkiene, R. (2020, December 11). *What is AES Encryption and How Does It Work?* CyberNews. https://cybernews.com/resources/what-is-aes-encryption/

Manico, J., & Detlefsen, A. (2015). *Iron-Clad Java : Building Secure Web Applications /*. New York, N.Y. Mcgraw-Hill Education.