****

# Practices for Secure Software Report

By: Charles Campbell

Table of Contents

[Document Revision History 3](#_Toc102040754)

[Client 3](#_Toc102040755)

[Instructions 3](#_Toc102040756)

[Developer 3](#_Toc102040757)

[1. Algorithm Cipher 4](#_Toc102040758)

[2. Certificate Generation 5](#_Toc102040759)

[3. Deploy Cipher 6](#_Toc102040760)

[4. Secure Communications 7](#_Toc102040761)

[5. Secondary Testing 8](#_Toc102040762)

[6. Functional Testing 10](#_Toc102040763)

[7. Summary 13](#_Toc102040764)

[8. Industry Standard Best Practices 13](#_Toc102040765)

[9. References 14](#_Toc102040765)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **Feb. 21, 2025** | **Charles Campbell** | **Final** |

## 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

Charles Campbell

## Algorithm Cipher

One of the top encryption tools [on Oracle’s cipher list](https://docs.oracle.com/javase/9/docs/specs/security/standard-names.html#cipher-algorithm-names), is the AES or Advanced Encryption Standard. Full details of this cipher are under the [government FIPS 197 document](https://csrc.nist.gov/files/pubs/fips/197/final/docs/fips-197.pdf). Developed in 1997 by the NIST, it is a cipher that the U.S. government both uses and recommends to encrypt and decrypt data; it utilizes a 256-bit key length (considered virtually uncrackable) and is the most secure algorithm in the industry (Kiteworks, n.d). Many banks and financial institutions also utilize it, which correlates with Artemis’s business functions and goals for modern and high security. It is still vulnerable to side-channel attacks, where hackers can pick up leaked information from non-secure systems to understand how the encryption algorithm works (Kiteworks, n.d). It’s recommended that Artemis also follow security practices like user validation and continuous monitoring to ensure that the cipher’s keys or patterns get leaked.

The AES cipher is also a symmetrical encryption tool, meaning it utilizes a single key to both encrypt and decrypt ciphertext. “Senders and recipients using symmetric encryption to transfer data to each other must know the secret key to, in the case of senders, encrypt the data they intend to share with recipients, and in the case of recipients, decrypt and read the encrypted data the senders share with them, as well as encrypt any necessary responses” (Daniel, 2021). AES utilizes this through taking a block of 256 bits of data and encrypting it. Through the same key, receivers can then decrypt that ciphertext to the original data. This algorithm takes the block of data into a 4-by-4 array, substitutes different (but pre-determined) bytes for another value, and then shifts each row a certain amount of spaces, and is then executed another 14 times causing a; the decryption then retraces those steps to result in the original message (Kiteworks, n.d). This algorithm cipher would take an estimated billion years of brute force to crack, which makes it one of the most successful and secure ciphers in the industry.

## Certificate Generation - Screenshot below of the CER file.

The below certificate was made through the terminal in the Eclipse IDE:

A screen shot of a computer

AI-generated content may be incorrect.

A computer screen with white text and colorful letters

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

These digital certificates help regulate clients by ensuring that authorized and trusted users only gain access via secure communication methods. These can be accomplished through the use of a certificate authority (CA), which create the cryptographic certificate and distribute them so that proper authoritative clients can show they can be trusted and helps maintain a secure overall system (Manico & Detlefsen, 2014). This advantageous tool helps software and web-based products to be shared through trusted and verified users, and adds a layer of security that deters and protects data from unauthorized and malicious individuals.

## Deploy Cipher - Screenshot below of the checksum verification.

A screen shot of a computer program

AI-generated content may be incorrect.

A black text on a white background

AI-generated content may be incorrect.

## Secure Communications - Screenshot below of the web browser that shows a secure webpage.

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

## Secondary Testing - Screenshots below of the refactored code executed without errors & the dependency-check report.

This is a screenshot of the dependency-check report before implementing any changes to the current codebase:  
A screenshot of a computer

AI-generated content may be incorrect.

These are the code changes I implemented to ensure proper connection to an HTTPS webpage:

A screen shot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.

The dependency-check is run again, and shows that the code I implemented generated no additional vulnerability errors within the application:

A screenshot of a computer

AI-generated content may be incorrect.

The vulnerable dependency’s match, indicating any additional or refactored code I implemented did not cause any errors or additional vulnerabilities within the program.

## Functional Testing - Screenshot below of the refactored code executed without errors.

Dependency check before functional testing & refactoring:

A screenshot of a computer

AI-generated content may be incorrect.

The dependency check results show that 142 vulnerabilities are found with 18 dependencies causing those vulnerabilities. One with the most critical severity is the Spring Framework dependency. In multiple suggestions from the dependency check, it’s advised to update to newer versions/releases of Spring as the developers fixed the issues causing the vulnerabilities.

The most up-to-date version of Spring that still runs on Java 1.8 is release: 2.7.18. Anything over requires newer versions of Java, otherwise it will not compile/run. Below is the refactored code in the pom.xml:

A screen shot of a computer program

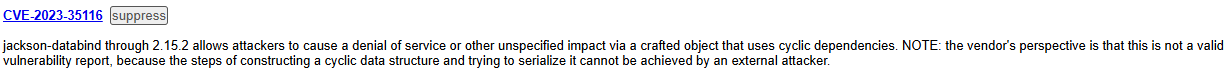
AI-generated content may be incorrect.

With this change, the resulting dependency check shows that 8 dependencies and 115 total vulnerabilities are resolved:

A screenshot of a computer

AI-generated content may be incorrect.

The remainder of these vulnerable dependencies are utilized by the old Spring 2.7.18 version. Some are false positives, as the dependency check shows in the first item reported, under the “Jackson-databind-2.13.5.jar”:



Suppression of these false positives can be implemented through the inclusion of a suppression.xml file in the project:

A screen shot of a computer program

AI-generated content may be incorrect. A screen shot of a computer program

AI-generated content may be incorrect.

Additionally, many of the other dependencies that still have vulnerabilities that aren’t false positives are utilized by the outdated Spring Framework. The only way to get rid of these dependencies would be to update to the newest 3.x version of Spring, but that requires also updating to the newest version of java as Spring 3.x cannot run on Java 1.8.

Spring’s developers have documented there will be no further updates to 2.x versions, “After 5.5 years and 121 releases, 2.7.18 marks the end of open source support for Spring Boot 2.x. Please upgrade to Spring Boot 3 as soon as possible” (Wilkinson, 2023).

It is recommended that Artemis Financial make this decision to have the most up-to-date Java & Spring versions, as this will not only further enhance the overall application to more modern operations, but also apply it’s vision to utilize the most current and effective software security.

Until that change is made, suppressions of these unfixable vulnerabilities & dependencies utilized by the out-of-date Spring version can be made to allow for a clean dependency-check report:

A screenshot of a computer

AI-generated content may be incorrect.

## Summary

Upon review with the Vulnerability Assessment Process Flow Diagram, Artemis Financial’s software application (being web-based) should rely on key security areas of cryptography, input validation and code quality. Cryptography is among the largest attack areas that hackers take advantage of, if it is improperly implemented in an application. For Artemis, in dealings with money, this is especially important to ensure that proper data/files are properly encrypted. “For example, passwords, credit card numbers, health records, personal information, and business secrets require extra protection, mainly if that data falls under privacy laws” (OWASP, n.d). Input validation also falls under this, as proper validation of input such as passwords (or ensuring that input passwords are strong enough) and confirming changes like digital money transfers or changes to financial plans can help deter cyber-attacks. The addition of the algorithm cipher and implementation of a certificate I added greatly increases the security of Artemis’s program, and fall under these key pillars of cryptography & input validation.

Code quality ensures that through each stage of the development cycle for software, security is present and accounted for, especially during the creation of the code. As Jeganathan suggests, “Apply and enforce secure-coding practices (OWASP & CERT) and integrate SAST tools [Secure Application Security Testing] like Veracode or OWASP SonarQube into IDE tools (Eclipse, Visual Studio, etc.)” (2019). These are important security practices that not only help reduce attack areas, but also are modern industry practices that all businesses and developers follow to ensure civilians and businesses alike stay safe from attackers.

## Industry Standard Best Practices

OWASP (a leading modern software security force) identifies in [their top 10 list](https://owasp.org/www-project-top-ten/) the major current & potential future issues that are the cause of external threats occurring on modern software systems and web applications. Specifically: vulnerable & outdated components (#2 on the list) can put software at risk of attackers manipulating data for DoS (denial of services) or gaining access to protected information; identification & authentication failures can lead to breaches of cybersecurity which allow for malicious modification and recording of private data through illegitimate access or posing as an authorized user, otherwise known as spoofing (OWASP, n.d). These are just a few examples, and awareness of these vulnerabilities & attack methods is the first step to combating external threats.

Through applying industry standard best practices, the code I implemented to assist the application be accessible via HTTPS, and the code that was refactored (updating the Spring plug-in to version 2.7.18) helped greatly improve Artemis’s software and ensured the company’s well-being. This was shown throughout the functional testing I implemented, where one simple update on an outdated component (Spring) helped eliminate a large portion of the vulnerabilities in Artemis’s application; this is a small example of something all software developers should do. “Developers should be trained on the secure coding techniques that are relevant for their project and technology stack, and on how to use the tools and methods … specified by security architects” (Manico, & Detlefsen, 2017). Following industry best practices for coding & security, Artemis (or any other company) can keep both itself and their customers safe. Continuous awareness and test of industry best practices can ensure everyone’s well-being is accounted for.

References

Daniel, B. (2024, October 21). *Symmetric vs. asymmetric encryption: What’s the difference?* Trusted Computing Innovator. <https://www.trentonsystems.com/en-us/resource-hub/blog/symmetric-vs-asymmetric-encryption>

*Everything you need to know about AES-256 encryption*. Kiteworks. (n.d.) <https://www.kiteworks.com/risk-compliance-glossary/aes-256-encryption/>

Federal Information Processing Standards. (2001, November 26). *ADVANCED ENCRYPTION STANDARD (AES).* <https://csrc.nist.gov/files/pubs/fips/197/final/docs/fips-197.pdf>

Jeganathan, S. (2019). DevSecOps: A Systemic Approach for Secure Software Development. *ISSA Journal*, *17*(11), 20–27.

Manico, J., & Detlefsen, A. (2014, September). *Iron-Clad Java: Building Secure Web Applications.* O’Reilly Online Learning*.* <https://www.oreilly.com/library/view/iron-clad-java/9780071835886/>

Oracle. (n.d.). *Java Security Standard Cipher Algorithm Names*. <https://docs.oracle.com/javase/9/docs/specs/security/standard-names.html#cipher-algorithm-names>

OWASP Foundation. *OWASP Top Ten*. (n.d.). <https://owasp.org/www-project-top-ten/>

Wilkinson, A. (2023, November 23). *Spring Blog*. Spring. <https://spring.io/blog/2023/11/23/spring-boot-2-7-18-available-now>