****

# 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** | **[2/16/23]** | **[Jared Smith]** | **Initial form fill** |

## 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

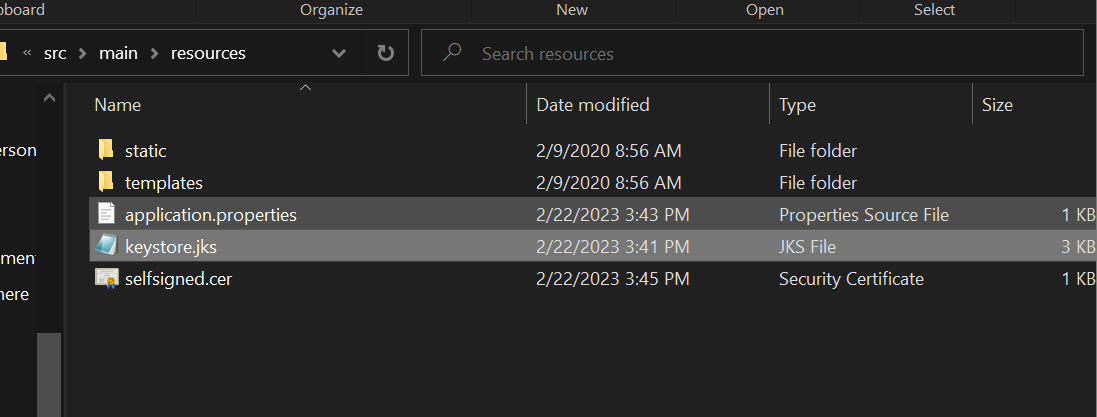
Jared Smith

## Algorithm Cipher

Artemis financial is looking for an encryption system for long term file storage. Because they are looking to store files instead of transfer them, and because speed is not a major concern, I am going to recommend they use the SHA-256 algorithm cypher. This cypher is great for long term storage for a few reasons. First, the cypher cannot be unhashed by conventional means. Because it’s a symmetric cypher, you need the original key to decrypt the files. If this key is kept secure at Artemis financial, the odds that bad actors are able to use any information they retrieve if they somehow break into the storage is pretty low. Secondly, SHA-256 has a very low collision probability, somewhere in the order of 10^-60. Collisions are duplicate results for different keys, which could cause vulnerabilities if malicious actors were able to predict a collision in the storage.

## Certificate Generation

Insert a screenshot below of the CER file.



## Deploy Cipher

Insert a screenshot below of the checksum verification.

Graphical user interface, text, application, email

Description automatically generated

## Secure Communications

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

Graphical user interface, text, application

Description automatically generated

## Secondary Testing

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

Text

Description automatically generated

Text

Description automatically generated

## Functional Testing

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

Text

Description automatically generated

## Summary

The code that was added for the algorithm cypher did not introduce any new dependencies with known vulnerabilities in the Maven dependency check. On manual review of the code, a few obvious flaws exist in the code at present. Because of how the code is set up for testing at the moment, valid CA keys are still self generated and will show as invalid until a list of valid certificates are either listed for the application or external CA’s are obtained for the application.

Lastly, input validation for the specific file types expected should be implemented, as most of the significant security vulnerabilities in Springboot and Tomcat exist with default configurations not having sufficient input validation. If the system will only be used on an internal network, self generated CA’s will be acceptable for this application. However, input validation should still be implemented to protect our server from potential attacks.

## Industry Standard Best Practices

As for industry security standards, we are currently using the industry standard cypher for file storage, and using common industry apis (Springboot and Tomcat) for our web application. Using well documented and industry standard apis and practices is an important and effective strategy for security for a few reasons. Firstly, we know the security vulnerabilities in our APIs and strategies, meaning we can mitigate those risks and address them in other parts of the code. Next, using well vetted strategies can give us assurance that most if not all security vulnerabilities have been identified in our codebase and it will not likely require significant refactoring any time soon. Lastly, if a new security vulnerability is identified in our codebase, we will likely know about the vulnerability quickly, and a solution to the risk will likely be discovered quickly.