****

# 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** | **12/5/2022** | **Ryan Niebla** |  |

## 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

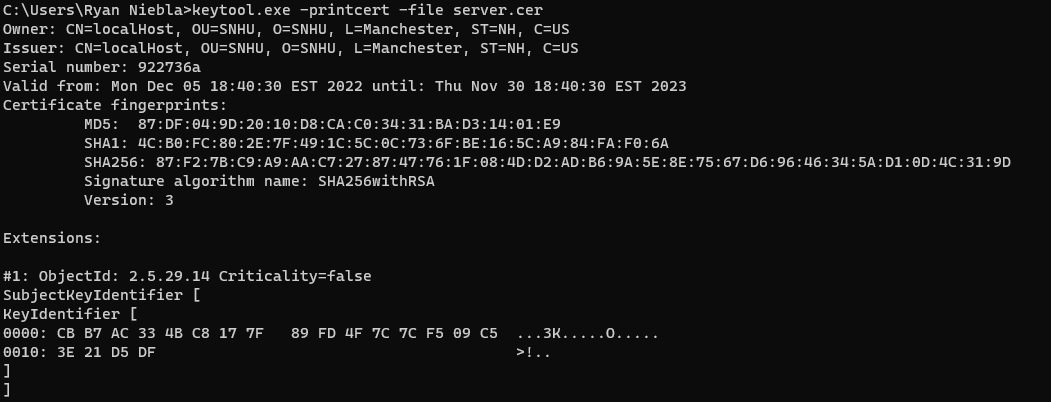
Ryan Niebla

## Algorithm Cipher

I implemented Secure Hash Algorithm (SHA)-256. I went with the 256 as opposed to higher because as you increase the hash size the complexity increases along side the processing time. Collisions are always a concern with hash functions. A collision occurs when multiple inputs yield the same hash value which presents large issues if you are storing something like a password. To date, SHA-256 has had no known collisions and is being widely used by many secure networks which is why I opted to used SHA-256 for our hashing needs.

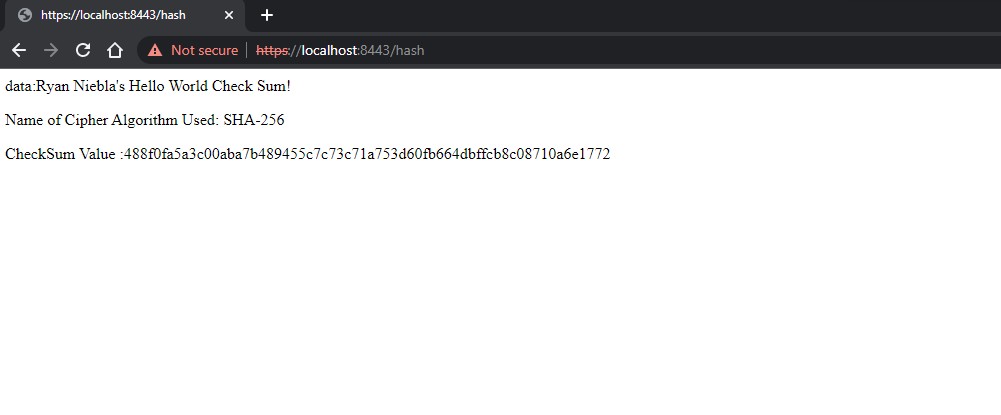
## Certificate Generation

Using Keytool I created then exported a self signed cert.

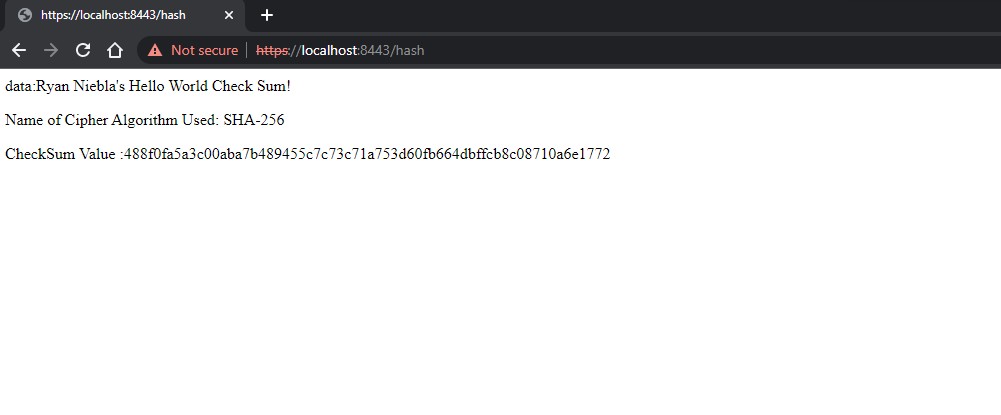


## Deploy Cipher

## The code was reformatted and deployed with the new cert to validate it ran

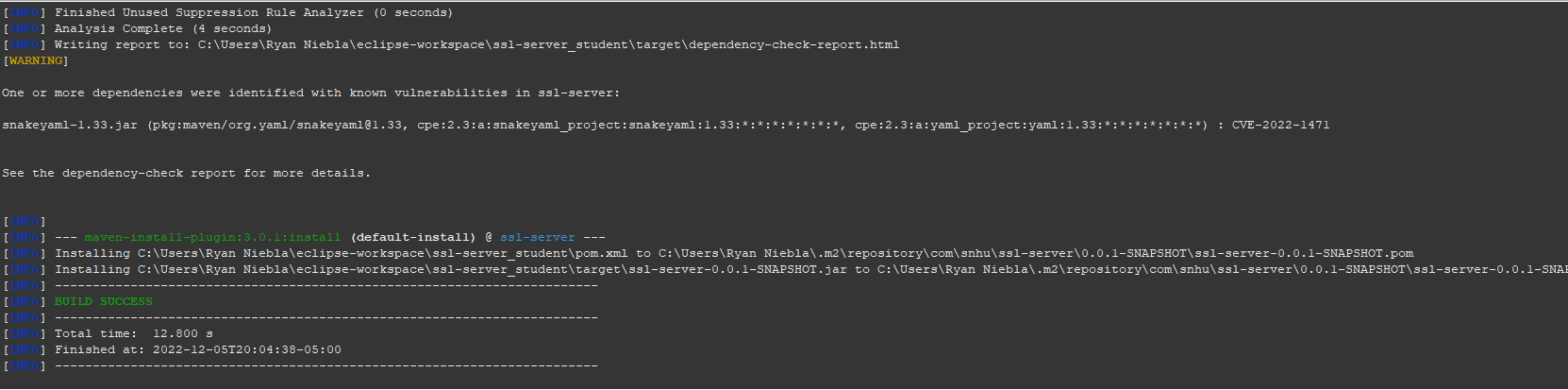


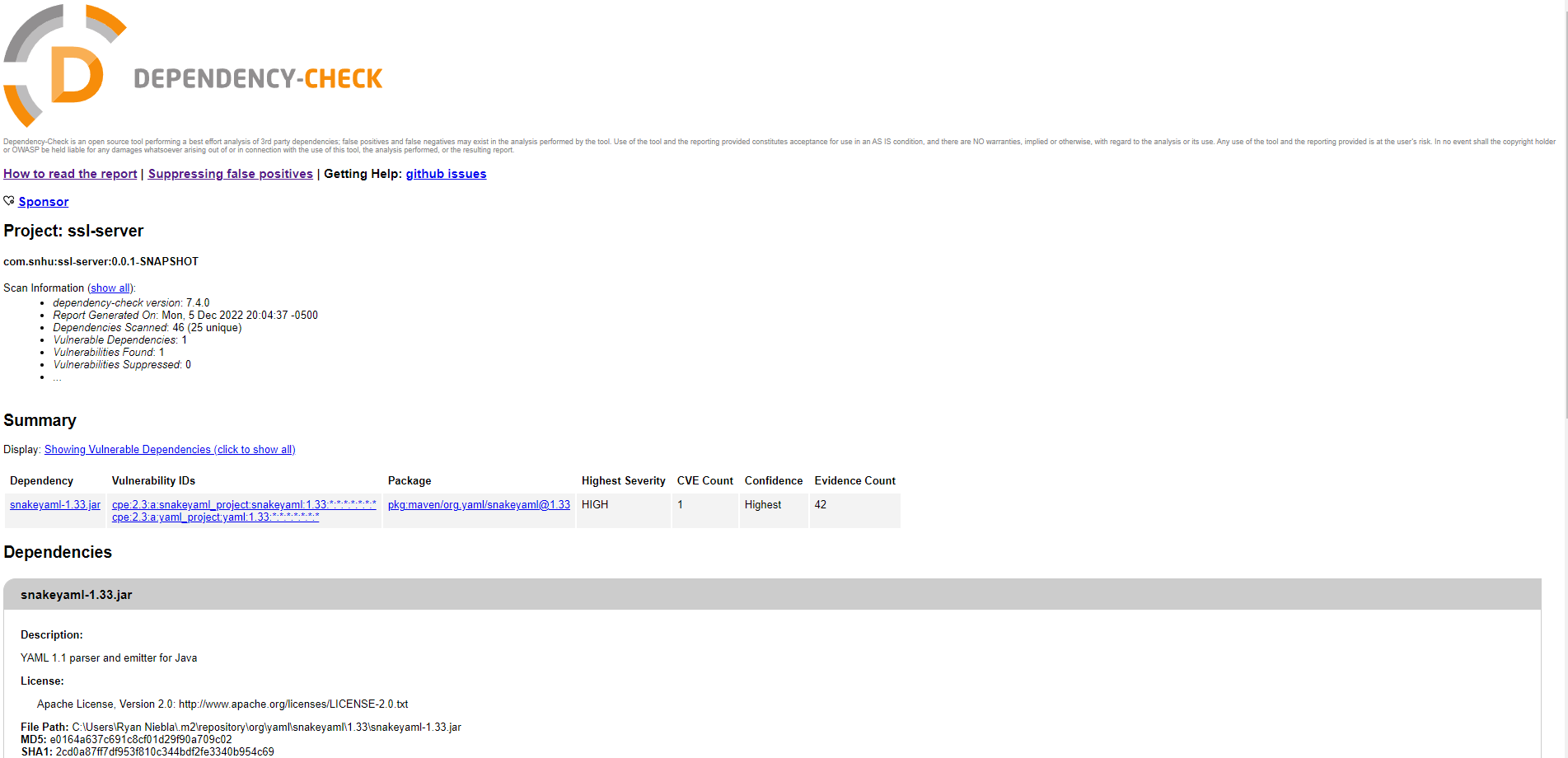
## Secure Communications

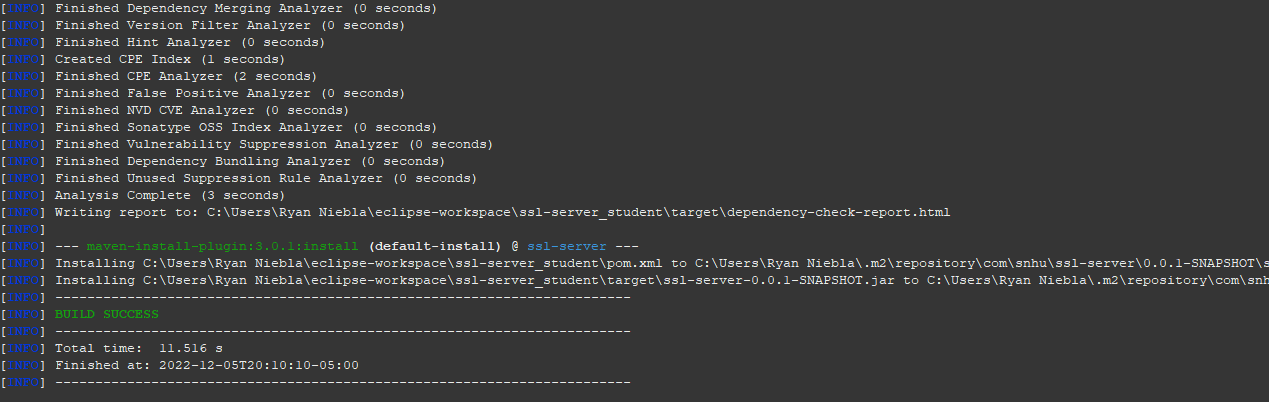


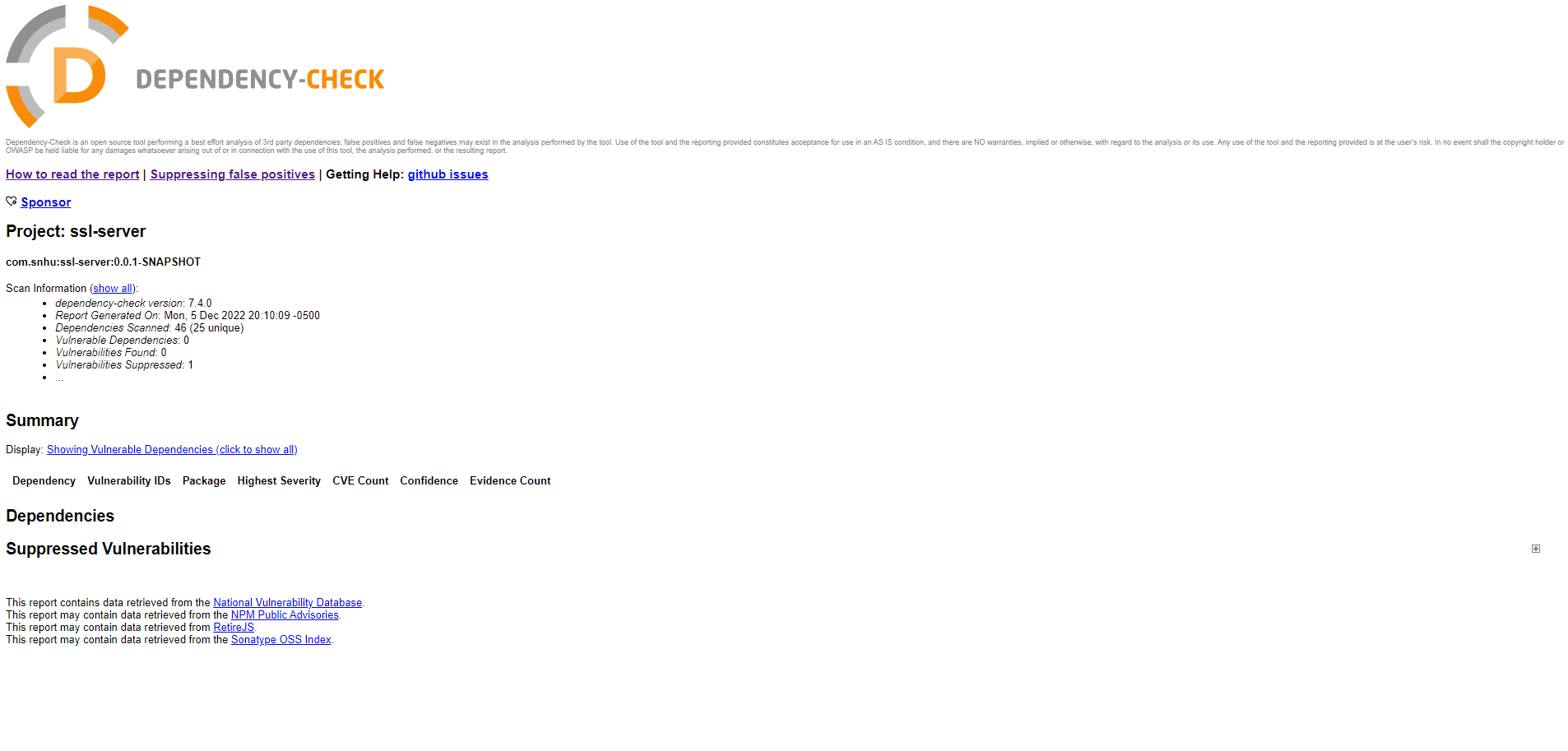
## Secondary Testing

## Showing my initial run at springBoot 2.7.4, then my latest upgrading to 3.0.0 and suppressing snakeYAML finding.





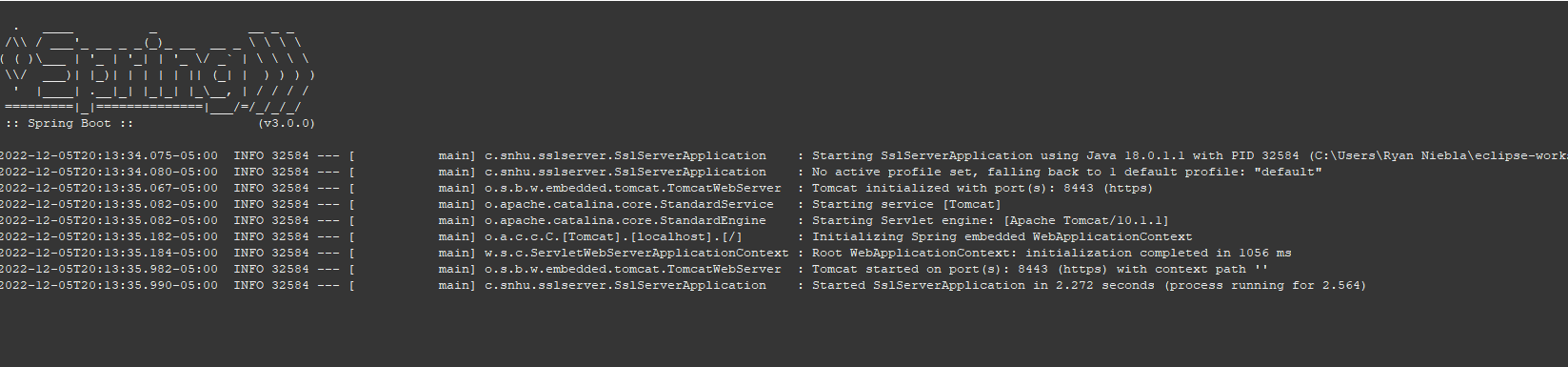




## Functional Testing

## Validated that all code smells were addressed and confirmed that the project runs

## 



## Summary

This wasn’t so much a refactor of code as there wasn’t a lot here to begin with. To address the Vulnerability assessment Process flow diagram. We worked with securing API’s through generating a certificate adding it to the keystore and project and visiting the site on https. We worked with cryptography by using a hash function to encode some data. We addressed code errors and code quality by running test and looking for and resolving code smells.

Adding security to this application comprised of several steps. First, I generated a certificate keytools and exported that cert. I thin pull the cert and the store into the project and reflected that on the applications.properties file. Once that was completes, I worked on a hash function which took my name and a string, hashed it and output the results. Once this appeared to be finished, I ran the program and visited localhost:8443/hash to see if the application loaded. Once I confirmed it did I went on to testing. Initially I was running an older version on Spring Boot, I upgraded to 3.0.0 to resolve some findings and suppressed snakeYAML.

## Industry Standard Best Practices

Security is of paramount concern to most companies, but within the financial sector it becomes even more urgent that code is written with security in mind. For this project I used the best practices that I know to create a secure system. I researched and implemented a hash function with no known collisions, I created and implemented a certificate for secure access via https and I tested my dependencies and updated when I saw issues. Security is a major concern as a developer and using the best methods we know to make the customer and the company more secure is obligatory in this field