GLOBAL RAIN

CS 305 Project Two
Practices for Secure Software Report

Table of Contents

Document Revision History

Version	Date	Author	Comments
1.0	2/20/2022	Enrique Zarate	

Client

Instructions

Deliver this completed Practices for Secure Software Report documenting your process for writing secure communications and refactoring code that complies with software security testing protocols. Respond to the steps outlined below and replace the bracketed text with your findings in your own words. If you choose to include images or supporting materials, be sure to insert them throughout.

Developer
Enrique Zarate

1. Algorithm Cipher
Determine an appropriate encryption algorithm cipher to deploy given the security vulnerabilities, justifying your reasoning. Be sure to address the following:

- Provide a brief, high-level overview of the encryption algorithm cipher.
- Discuss the hash functions and bit levels of the cipher.
- Explain the use of random numbers, symmetric vs non-symmetric keys, and so on.
- Describe the history and current state of encryption algorithms.

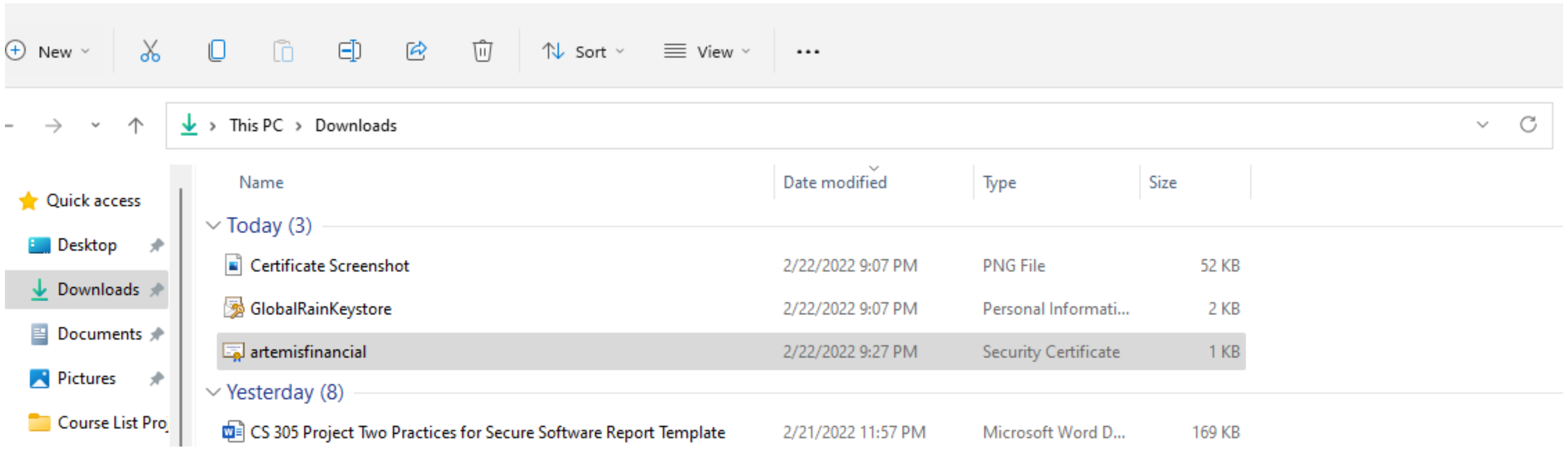
For this project and encrypting communications going in and out of the primary webpage application, assessing the potential information being sent through the webface and the likelihood of cyber attacks being increased because the facility is a financial institution, I'm recommending that we use the higher end of the encryption and take it a step above AES and use RSA. The RSA encryption algorithm allows for key length between 1024 and 4096 bits which will provide the stronger protection we need to better protect our customers and their information. The longer length of bits will allow for higher variability in randomness decreasing the chances of allowing hackers to break the code.

The RSA encryption also uses the non-symmetric key system which will provide two different keys for encryption and decryption of data. This will allow for a public key which will be able to encrypt any data being sent over from the customer to the organization and only those with the decryption key will be able to read the data. This will provide better encryption for hackers trying to receive the transmissions over the web and even if they were to retrieve the information, they would need a separate key to open the data and be able to see it.

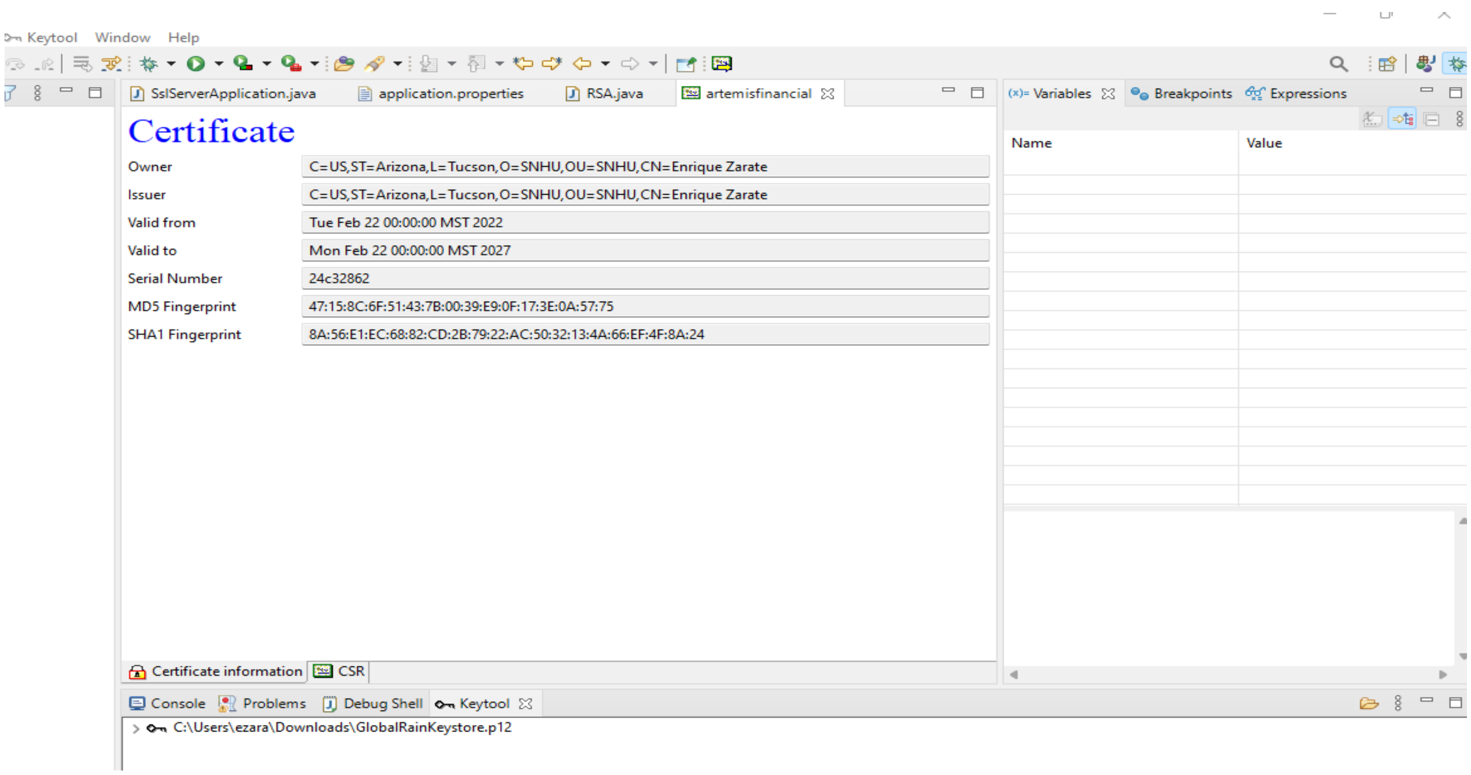
Cybersecurity if an ever-changing space and the more technology advances, whether it is advances in hacking techniques or advances in security properties, those in the field need to keep up with those changes to continue to protect the data of their consumers from being misused.

2. Certificate Generation
Generate appropriate self-signed certificates using the Java Keytool, which is used through the command line.

- To demonstrate that the keys were effectively generated, export your certificates (CER file) and submit a screenshot of the CER file below.



Name	Date modified	Type	Size
Today (3)			
Certificate Screenshot	2/22/2022 9:07 PM	PNG File	52 KB
GlobalRainKeystore	2/22/2022 9:07 PM	Personal Informati...	2 KB
artemisfinancial	2/22/2022 9:27 PM	Security Certificate	1 KB
Yesterday (8)			
CS 305 Project Two Practices for Secure Software Report Template	2/21/2022 11:57 PM	Microsoft Word D...	169 KB



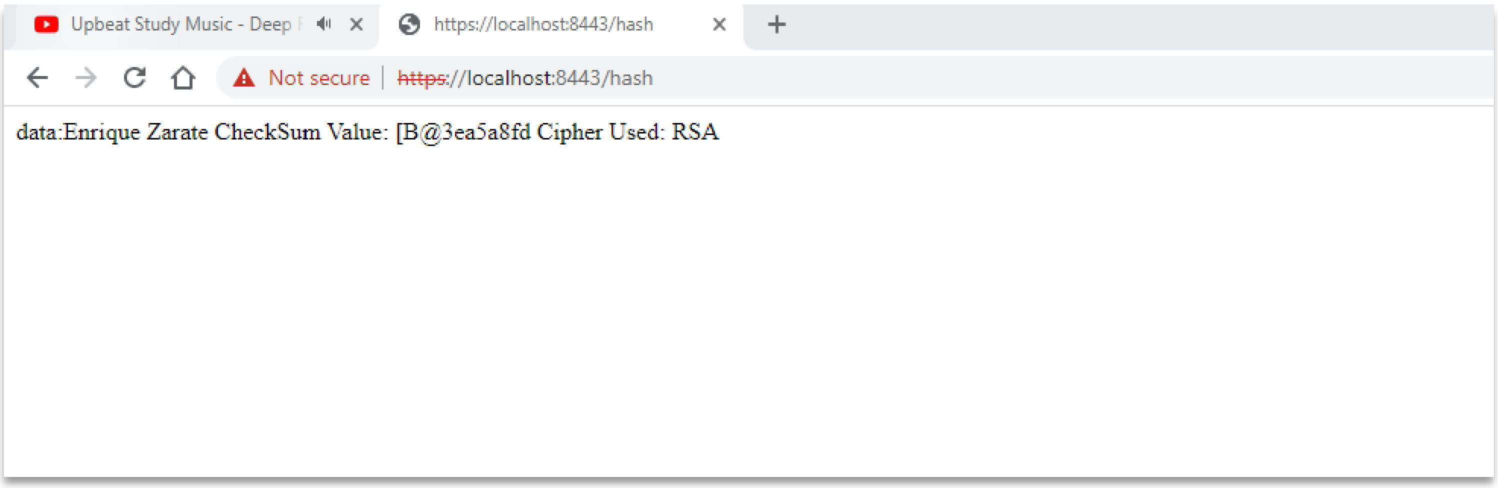
Certificate

Owner	C=US,ST=Arizona,L=Tucson,O=SNHU,OU=SNHU,CN=Enrique Zarate
Issuer	C=US,ST=Arizona,L=Tucson,O=SNHU,OU=SNHU,CN=Enrique Zarate
Valid from	Tue Feb 22 00:00:00 MST 2022
Valid to	Mon Feb 22 00:00:00 MST 2027
Serial Number	24c32862
MD5 Fingerprint	47:15:8C:6F:51:43:7B:00:39:E9:0F:17:3E:0A:57:75
SHA1 Fingerprint	8A:56:E1:EC:68:82:CD:2B:79:22:AC:50:32:13:4A:66:EF:4F:8A:24

```
C:\Users\ezara\Downloads\GlobalRainKeystore.p12
```

3. Deploy Cipher
Refactor the code and use security libraries to deploy and implement the encryption algorithm cipher to the software application. Verify this additional functionality with a checksum.

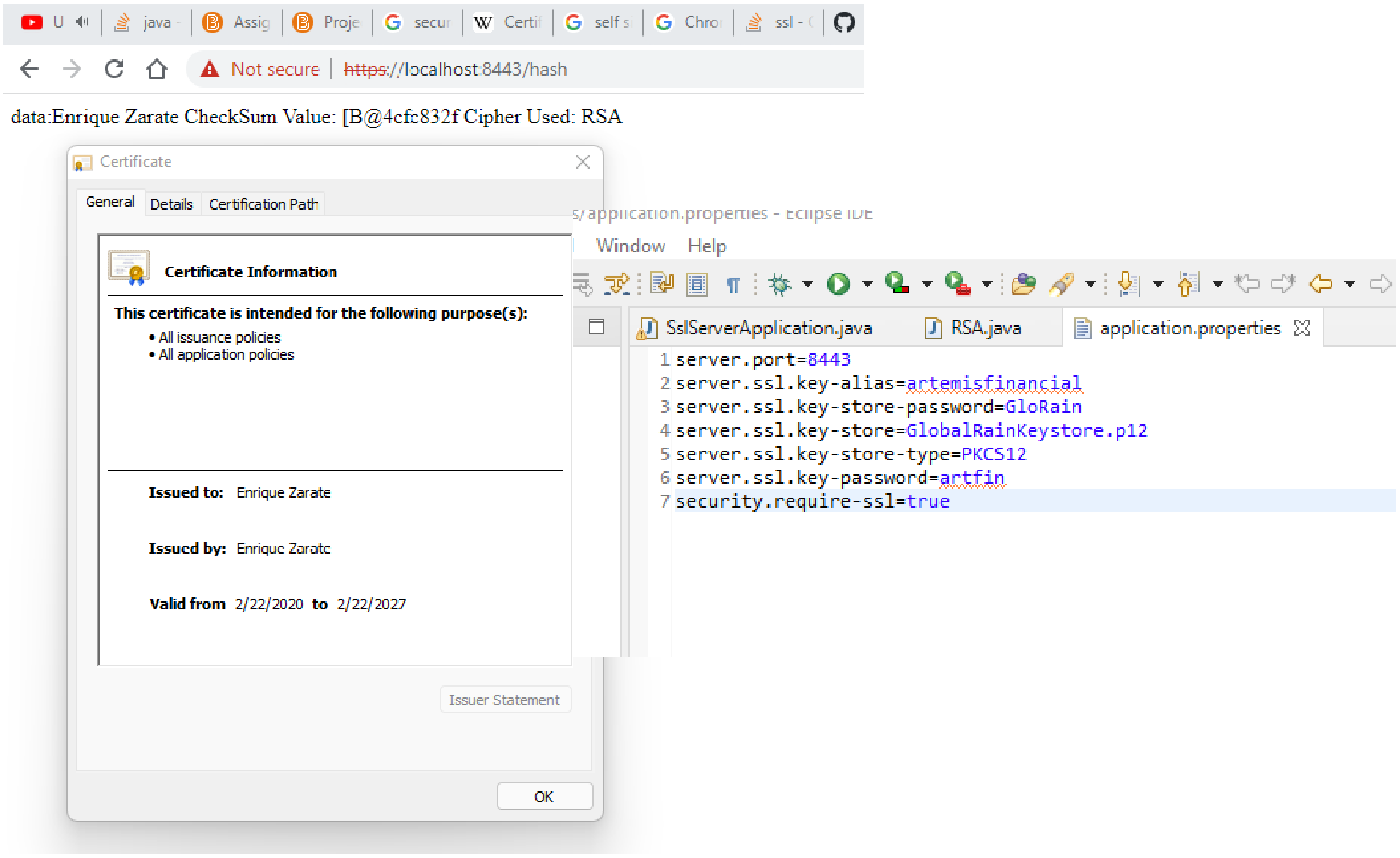
- Insert a screenshot below of the checksum verification. The screenshot must show your name and a unique data string that has been created.


Upbeat Study Music - Deep F
https://localhost:8443/hash
Not secure
~~https~~://localhost:8443/hash

```
data:Enrique Zarate CheckSum Value: [B@3ea5a8fd Cipher Used: RSA
```

4. Secure Communications
Refactor the code to convert HTTP to the HTTPS protocol. Compile and run the refactored code to verify secure communication by typing **https://localhost:8443/hash** in a new browser window to demonstrate that the secure communication works successfully.

- Insert a screenshot below of the web browser that shows a secure webpage.
- Please note… Chrome and many browsers have made it so that using your own Signed Certificates, even though installing the certificate on your own machine, will still show the warning unless further modifications are made in the background. Providing two pictures that show the site has a certificate and that the code is placed in correctly.



data:Enrique Zarate CheckSum Value: [B@4cfc832f Cipher Used: RSA

Certificate

Certificate Information

This certificate is intended for the following purpose(s):
- All issuance policies
- All application policies

Issued to: Enrique Zarate

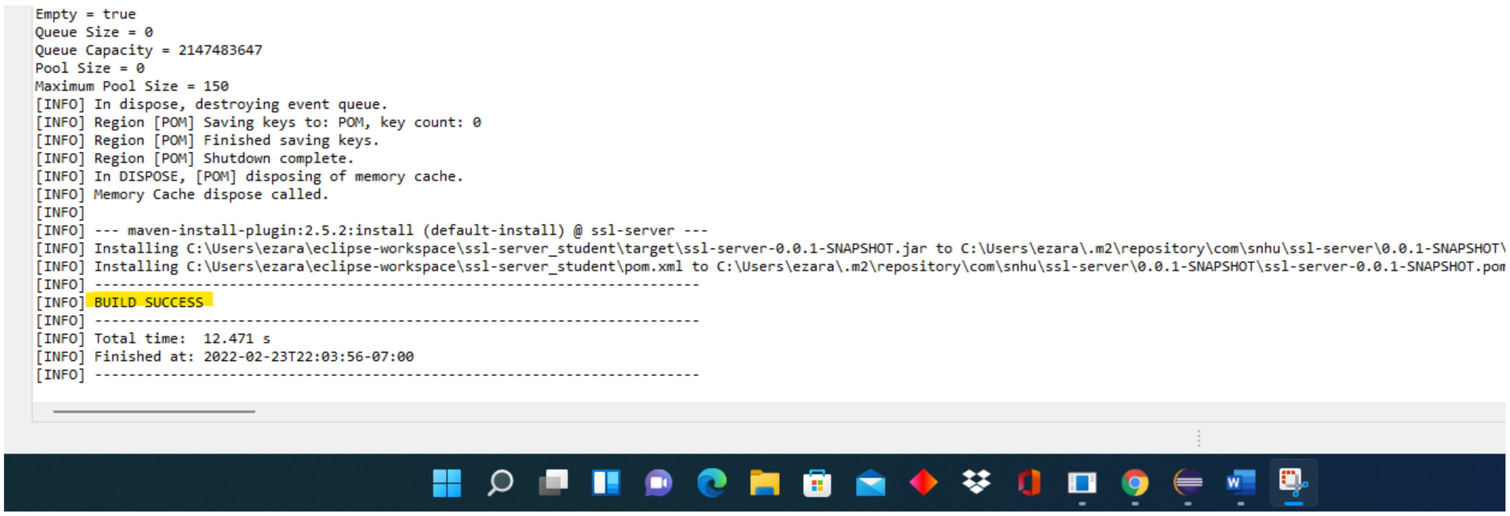
Issued by: Enrique Zarate

Valid from 2/22/2020 to 2/22/2027

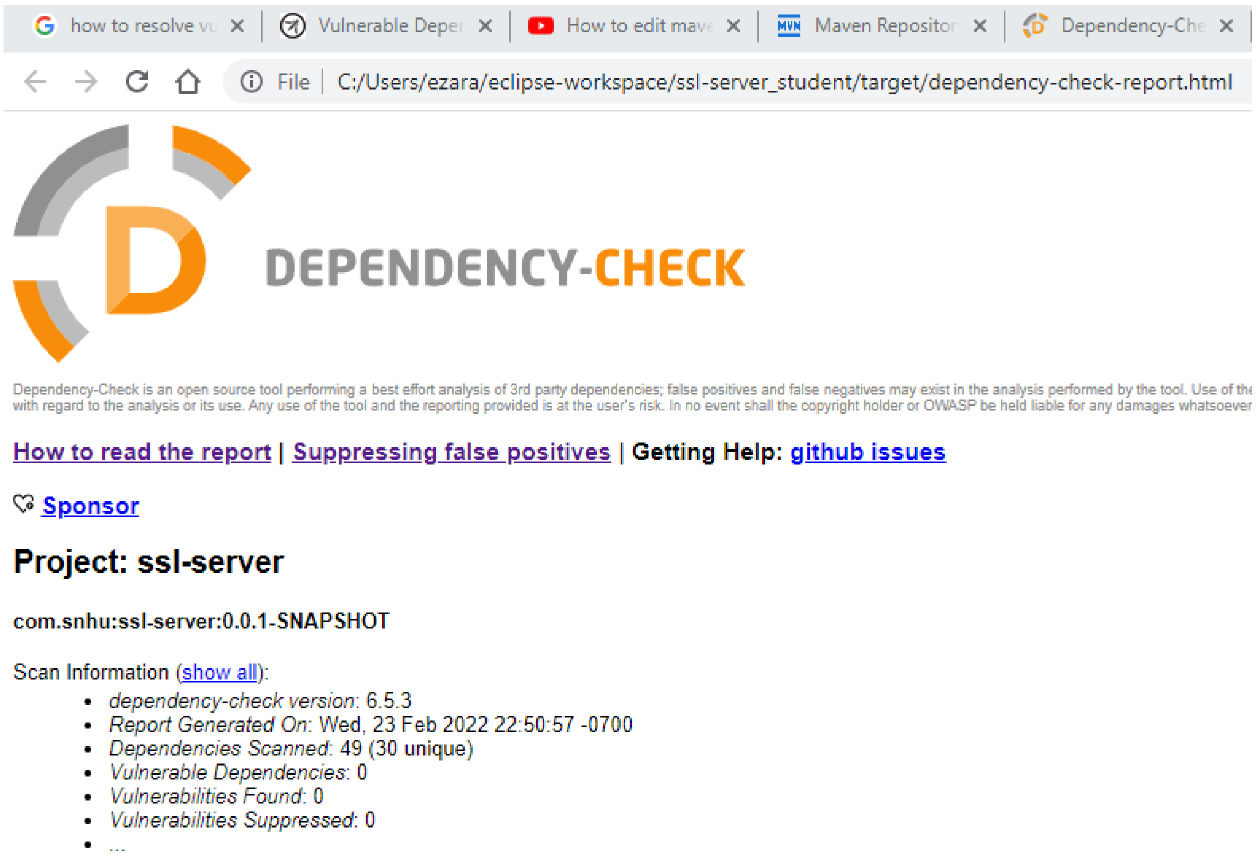
```
server.port=8443
server.ssl.key-alias=artemisfinancial
server.ssl.key-store-password=GloRain
server.ssl.key-store=GlobalRainKeystore.p12
server.ssl.key-store-type=PKCS12
server.ssl.key-password=artfin
security.require-ssl=true
```

5. Secondary Testing
Complete a secondary static testing of the refactored code using the dependency check tool to ensure code complies with software security enhancements. You only need to focus on the code you have added as part of the refactoring. Complete the dependency check and review the output to ensure you did not introduce additional security vulnerabilities.

- Include the following below:
  - A screenshot of the refactored code executed without errors
  - A screenshot of the dependency check report



```
Empty = true
Queue Size = 0
Queue Capacity = 2147483647
Pool Size = 0
Maximum Pool Size = 150
[INFO] In dispose, destroying event queue.
[INFO] Region [POM] Saving keys to: POM, key count: 0
[INFO] Region [POM] Finished saving keys.
[INFO] Region [POM] Shutdown complete.
[INFO] In DISPOSE, [POM] disposing of memory cache.
[INFO] Memory Cache dispose called.
[INFO]
[INFO] --- maven-install-plugin:2.5.2:install (default-install) @ ssl-server ---
[INFO] Installing C:\Users\ezara\eclipse-workspace\ssl-server_student\target\ssl-server-0.0.1-SNAPSHOT.jar to C:\Users\ezara\.m2\repository\com\snhu\ssl-server\0.0.1-SNAPSHOT\
[INFO] Installing C:\Users\ezara\eclipse-workspace\ssl-server_student\pom.xml to C:\Users\ezara\.m2\repository\com\snhu\ssl-server\0.0.1-SNAPSHOT\ssl-server-0.0.1-SNAPSHOT.pom
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  12.471 s
[INFO] Finished at: 2022-02-23T22:03:56-07:00
[INFO] ------------------------------------------------------------------------
```



Dependency-Check is an open source tool performing a best effort analysis of 3rd party dependencies; false positives and false negatives may exist in the analysis performed by the tool. Use of the
with regard to the analysis or its use. Any use of the tool and the reporting provided is at the user's risk. In no event shall the copyright holder or OWASP be held liable for any damages whatsoever

How to read the report | Suppressing false positives | Getting Help: github issues

Sponsor

Project: ssl-server

com.snhu:ssl-server:0.0.1-SNAPSHOT

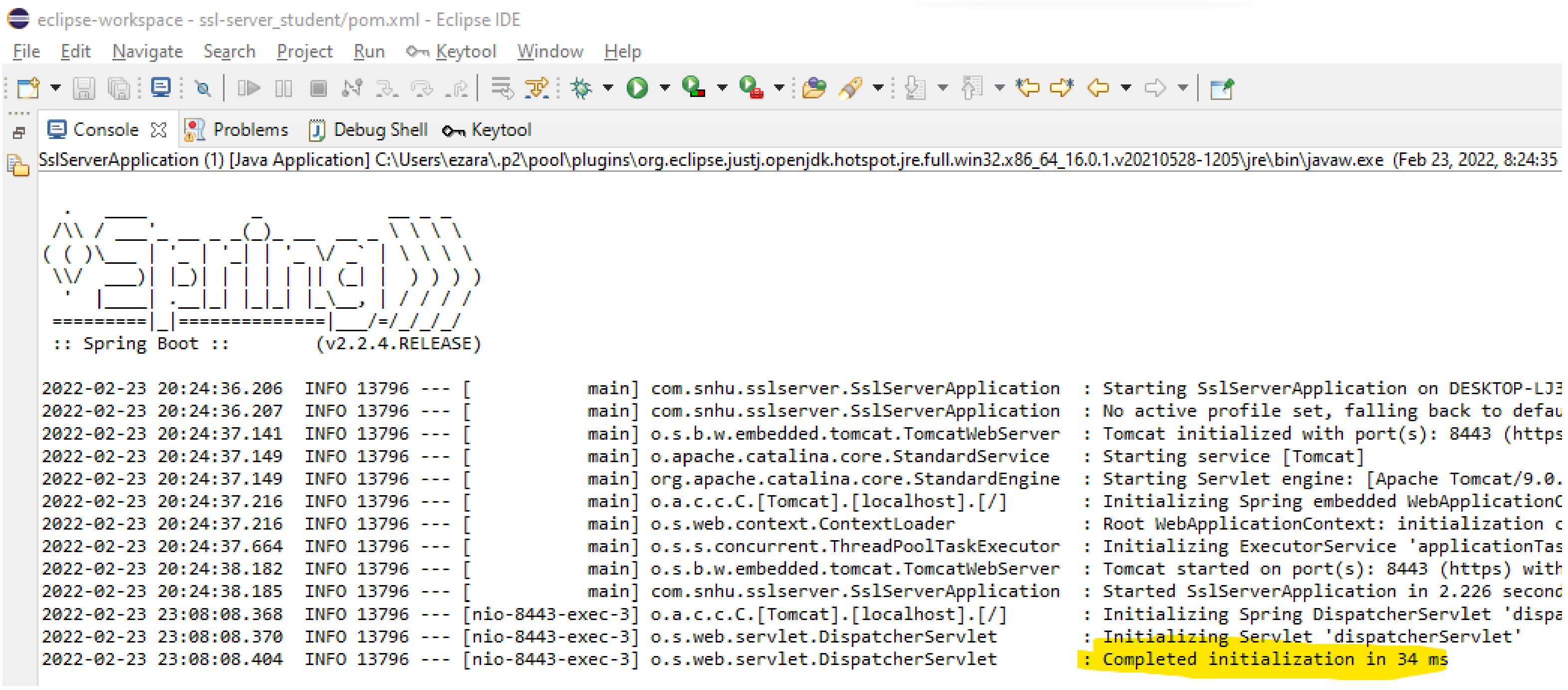
Scan Information (show all):

- *dependency-check version*: 6.5.3
- *Report Generated On*: Wed, 23 Feb 2022 22:50:57 -0700
- *Dependencies Scanned*: 49 (30 unique)
- *Vulnerable Dependencies*: 0
- *Vulnerabilities Found*: 0
- *Vulnerabilities Suppressed*: 0
- ...

6. Functional Testing

Identify syntactical, logical, and security vulnerabilities for the software application by manually reviewing code.

- Complete this functional testing and include a screenshot below of the refactored code executed without errors.



eclipse-workspace - ssl-server_student/pom.xml - Eclipse IDE
File Edit Navigate Search Project Run Keytool Window Help
Console Problems Debug Shell Keytool

```
SslServerApplication (1) [Java Application] C:\Users\ezara\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_16.0.1.v20210528-1205\jre\bin\javaw.exe  (Feb 23, 2022, 8:24:35

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::        (v2.2.4.RELEASE)

2022-02-23 20:24:36.206  INFO 13796 --- [           main] com.snhu.sslserver.SslServerApplication  : Starting SslServerApplication on DESKTOP-LJ3
2022-02-23 20:24:36.207  INFO 13796 --- [           main] com.snhu.sslserver.SslServerApplication  : No active profile set, falling back to defau
2022-02-23 20:24:37.141  INFO 13796 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8443 (https
2022-02-23 20:24:37.149  INFO 13796 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2022-02-23 20:24:37.149  INFO 13796 --- [           main] org.apache.catalina.core.StandardEngine  : Starting Servlet engine: [Apache Tomcat/9.0.
2022-02-23 20:24:37.216  INFO 13796 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationC
2022-02-23 20:24:37.216  INFO 13796 --- [           main] o.s.web.context.ContextLoader            : Root WebApplicationContext: initialization c
2022-02-23 20:24:37.664  INFO 13796 --- [           main] o.s.s.concurrent.ThreadPoolTaskExecutor  : Initializing ExecutorService 'applicationTas
2022-02-23 20:24:38.182  INFO 13796 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8443 (https) with
2022-02-23 20:24:38.185  INFO 13796 --- [           main] com.snhu.sslserver.SslServerApplication  : Started SslServerApplication in 2.226 second
2022-02-23 23:08:08.368  INFO 13796 --- [nio-8443-exec-3] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring DispatcherServlet 'dispa
2022-02-23 23:08:08.370  INFO 13796 --- [nio-8443-exec-3] o.s.web.servlet.DispatcherServlet        : Initializing Servlet 'dispatcherServlet'
2022-02-23 23:08:08.404  INFO 13796 --- [nio-8443-exec-3] o.s.web.servlet.DispatcherServlet        : Completed initialization in 34 ms
```

7. Summary

Discuss how the code has been refactored and how it complies with security testing protocols. Be sure to address the following:

- Refer to the Vulnerability Assessment Process Flow Diagram and highlight the areas of security that you addressed by refactoring the code.
- Discuss your process for adding layers of security to the software application and the value that security adds to the company's overall wellbeing.
- Point out best practices for maintaining the current security of the software application to your customer.

At this time, our focus has been on creating a secure connection between Artemis Financial Websites front page and the users. So our focus is primarily on encryption of data sent back and forth between the site and the customer as well as security layers to ensure secure transmission between the two. One of the primary ways we have done this is by adding a code of encryption into the system which will encrypt and decrypt all messages sent back and forth using an RSA algorithm. This should be a stronger and more suitable algorithm for this use as it different message will be encrypted randomly each time it is encrypted. This will decrease the chances that hackers may be able to crack the code which unlike a hex code, will stay the same until it is time to change or update the algorithm. The downside to this, is that it is harder to monitor when items are tempered with or corrupted since they are not the same most of the time.

The second part of this was creating a secure connection between the customer and our business. By obtaining a certificate and adding our credibility to the world wide web, we are notifying our customers that our website is a legit website that can be vouched for by outside entities that can back our website and ensure who we are. Along with securing the transmission of data between the two places and keeping data from being retrieved during transmission. In any event, if it were received, it would hopefully have a strong enough encryption that hackers would not be able to decrypt the data and would have useless data.

Lastly, we ran multiple dependency checks and updates the dependencies that had numerous vulnerabilities associated with them. By updating these dependencies, we are closing any gaps that hackers may have used in the past to bypass some security functions. This concludes the report and if you have any questions, feel free to reach out to the Global Rain for any questions.

A file titled “ssl-server_student Project 2 Refactored.zip” is being attached in this email so you can further observe the refactored code.