



GLOBAL RAIN

Practices for Secure Software Report

Table of Contents

DOCUMENT REVISION HISTORY	4
CLIENT	4
INSTRUCTIONS	4
DEVELOPER.....	ERROR! BOOKMARK NOT DEFINED.
1. ALGORITHM CIPHER	5
2. CERTIFICATE GENERATION	6
3. DEPLOY CIPHER.....	6
4. SECURE COMMUNICATIONS	6
5. SECONDARY TESTING.....	7
6. FUNCTIONAL TESTING	15
7. SUMMARY	18
8. INDUSTRY STANDARD BEST PRACTICES	19

General Hints if you have issue with refactoring from one of your peers:

Justin Starr justin.starr@snhu.edu

I do not believe you need to update the spring-boot version. The assignment is focused on whether or not your refactored code produces additional vulnerabilities. The only thing that I updated on the .pom file was the OWASP plugin version from 5.3.0 to 8.4.0. I did a dependency-check-report before and after refactoring the code.

The assignment is focused on whether or not your refactored code produces additional vulnerabilities. The only thing that I updated on the .pom file was the OWASP plugin version from 5.3.0 to 8.4.0. I did a dependency-check-report before and after refactoring the code.

However, concerning the .pom file, I will say that for the sake of trying, I updated the spring-boot version to 3.1.4 and received the same error you did. I did also clear my system cache and continued to receive the same error. I played around with it and found that I was able to successfully update the version to 3.0.8 without error. This does significantly reduce the number of vulnerabilities.

When I update the .pom file with spring-boot version 3.1.4, I am still able to run the SslServerApplication. It does load Spring Boot version 3.1.4. This only does not work for me when I try to get a dependency check report.

Document Revision History

Version	Date	Author	Comments
1.0	xxx	xxx	username: xxx password: xxx

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.

1. Algorithm Cipher

The client, Artemis Financial, requires a form of cryptography that can validate the integrity of the data being transferred. This can be achieved using a checksum. A checksum, or hash, is an algorithm that computes a unique hash value to verify the integrity of data while stored or in transit (Siebert, 2021). Commonly, cipher algorithms are thought to be encoded and decoded as seen on television. These types of algorithms typically use random numbers inserted into the data to increase the encryption strength of the cipher. To encode and decode data, these algorithms use keys of both symmetric and non-symmetric types. Symmetric keys use the same key to encode and decode the data. Non-symmetric keys, by contrast, use one key to encrypt data and a separate key to decrypt data. This is often referred to as a public-private key pair. However, the algorithm used for a checksum is a hashing algorithm, which is a one-way conversion of data into a random bit string. The hashing algorithm chosen must be secure by regulatory standards to produce an uncompromisable checksum. To be considered secure, the algorithm must not allow a message to be identified from only a checksum and must produce a unique checksum for every message (National Institute of Standards and Technology [NIST], 2015, August). The latter property is also known as the collision resistance of an algorithm. This property is important because it is wholly responsible for verifying data integrity. Without collision resistance, it would be impossible to verify the integrity of data from a checksum. The recommended algorithm for the functionality required is the Secure Hashing Algorithm (SHA), specifically SHA-256. SHA-256 is part of a family of hashing algorithms called SHA-2 with algorithms being separated by bit level. The bit level, represented by 256 in SHA-256, quantifies the length of the hash by the number of bits. These algorithms use the Boolean Choose function and Majority function in addition to a circular rotation of the bits to create secure hashes. The original hash function SHA-1, created in 1995, is no longer secure due to the increased

processing power of modern computers and should be eliminated by 2031 (NIST, 2022, December 15). SHA-256 and other members of the SHA-2 family of algorithms are still secure and sanctioned for use by the NIST. Therefore, the ideal algorithm for this functionality is SHA-256.

2. Certificate Generation

Insert a screenshot below of the CER file.

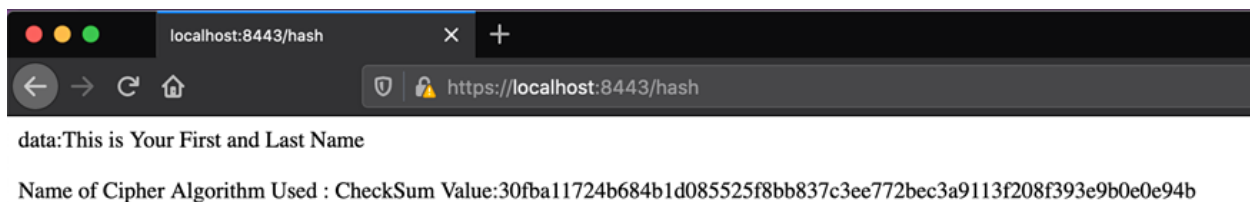
```
C:\Users\ax2707rx\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_19.0.2.v20230129-1123\jre>keytool -printcert -file proj3server.cer
Owner: CN=localhost, OU=best unit, O=best org, L=charlotte, ST=nc, C=us
Issuer: CN=localhost, OU=best unit, O=best org, L=charlotte, ST=nc, C=us
Serial number: a1ca6d0
Valid from: Tue Apr 11 14:07:44 EDT 2023 until: Fri Apr 05 14:07:44 EDT 2024
Certificate fingerprints:
    MD5: 52:B6:A8:C4:2A:97:32:02:D8:EF:D1:5B:1B:88:2E:23
    SHA1: 53:5C:F8:62:22:28:AD:EE:0F:48:59:D7:2D:EC:BC:4B:88:C5:E0:7F
    SHA256: 18:75:F7:5D:31:B0:59:6F:1E:C1:7F:9B:4C:BD:D0:80:F3:54:33:52:18:0F:C3:64:0E:46:6E:62:20:8A:73:E6
Signature algorithm name: SHA256withRSA
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 53 94 48 85 A8 11 5F 7B C8 D9 E1 35 2F 3E 47 CB S.H..._....5/>G.
0010: 3D 47 4A 08 =GJ.
]
]
```

3. Deploy Cipher

Insert a screenshot below of the checksum verification.



data:This is Your First and Last Name

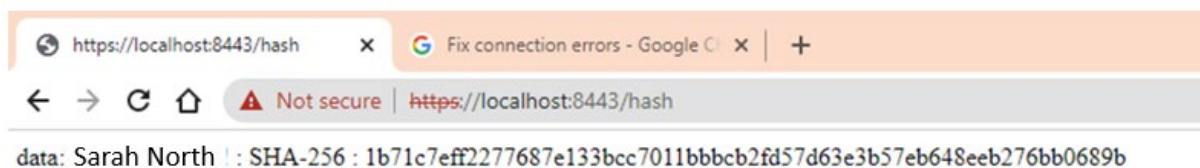
Name of Cipher Algorithm Used : CheckSum Value:30fba11724b684b1d085525f8bb837c3ee772bec3a9113f208f393e9b0e0e94b

4. Secure Communications

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



The browser will continue to say not secure as long as a self-signed certificate is used. This will be removed once the certificate used is from a CA.



5. Secondary Testing

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

SslServerApplication.java

```
6 import org.springframework.web.bind.annotation.RequestMapping;
7 import org.springframework.web.bind.annotation.RestController;
8 import java.security.MessageDigest;
9 import java.security.NoSuchAlgorithmException;
```

```

21 @RestController
22 class ServerController {
23
24
25     // Maps a route to the check sum
26     @RequestMapping(value="/hash")
27     final public String checkSumPage() {
28         final String data = "Jesse Fjestad";
29         final String checksum = getSHA(data);
30         return "data: " + data + "<br>" + "Checksum hash: " + checksum;
31     }
32
33     // Converts a string of data into a secure hexadecimal hash
34     private final static String getSHA(String data) {
35         MessageDigest mainDigest;
36         String secureHash = "";
37         try {
38             mainDigest = MessageDigest.getInstance("SHA-256");
39             mainDigest.update(data.getBytes());
40             byte[] digest = mainDigest.digest();
41             secureHash = bytesToHex(digest);
42         } catch (NoSuchAlgorithmException e) {
43             secureHash = "Could not identify algorithm.";
44         }
45         return secureHash;
46     }
47
48     // Converts a byte array into a hexadecimal string
49     private final static String bytesToHex(byte[] bytes) {
50         char[] hexArray = "0123456789abcdef".toCharArray();
51         char[] hexChars = new char[bytes.length * 2];
52         for ( int j = 0; j < bytes.length; j++ ) {
53             int v = bytes[j] & 0xFF;
54             hexChars[j * 2] = hexArray[v >>> 4];
55             hexChars[j * 2 + 1] = hexArray[v & 0x0F];
56         }
57         return new String(hexChars);
58     }
59 }
60

```

application.properties

```

2 server.port=8443
3 server.ssl.key-alias=proj3sig
4 server.ssl.key-store-password=mothergoose
5 server.ssl.key-store=src/main/resources/keystore/keystore.jks
6 server.ssl.key-store-type=JKS
7 server.ssl.enabled=true
8
9

```

pom.xml


```

61<⊖      <configuration>
62<⊖          <suppressionFiles>
63          <suppressionFile>suppression.xml</suppressionFile>
64          </suppressionFiles>
65      </configuration>

```

suppression.xml

Suppresses vulnerabilities in the original code before refactoring began.

```

1  <?xml version="1.0" encoding="UTF-8"?>
    Bind to grammar/schema...
2  <suppressions xmlns="https://jeremylong.github.io/DependencyCheck/dependency-suppression.1.3.xsd">

3      <suppress>
4          <notes><![CDATA[
5              file name: spring-data-rest-webmvc-3.2.4.RELEASE.jar
6          ]]></notes>
7          <packageUrl regex="true">^pkg:maven/org\.springframework\.data/spring\-data\-rest\-webmvc@.*$</p
8          <cve>CVE-2021-22047</cve>
9          <cve>CVE-2022-31679</cve>
10     </suppress>
11     <suppress>
12         <notes>
13         <![CDATA[
14             file name: bcprov-jdk15on-1.46.jar
15         ]]>
16         </notes>
17         <packageUrl regex="true">^pkg:maven/org\.bouncycastle/bcprov\-jdk15on@.*$</packageUrl>
18         <cve>CVE-2016-1000338</cve>
19         <cve>CVE-2016-1000342</cve>
20         <cve>CVE-2016-1000343</cve>
21         <cve>CVE-2016-1000344</cve>
22         <cve>CVE-2016-1000352</cve>
23         <cve>CVE-2016-1000341</cve>
24         <cve>CVE-2016-1000345</cve>
25         <cve>CVE-2017-13098</cve>
26         <cve>CVE-2020-15522</cve>
27         <cve>CVE-2020-0187</cve>
28         <cve>CVE-2016-1000339</cve>
29         <cve>CVE-2020-26939</cve>
30         <cve>CVE-2015-7940</cve>
31         <cve>CVE-2018-5382</cve>
32         <cve>CVE-2013-1624</cve>
33         <cve>CVE-2016-1000346</cve>
34         <cve>CVE-2015-6644</cve>
35     </suppress>
36     <suppress>
37         <notes>

```

```

38         <![CDATA[
39         file name: hibernate-validator-6.0.18.Final.jar
40         ]]>
41     </notes>
42     <packageUrl regex="true">^pkg:maven/org\.hibernate\.validator/hibernate\-validator@.*$</packageUrl>
43         <cve>CVE-2020-10693</cve>
44 </suppress>
45 <suppress>
46     <notes>
47         <![CDATA[
48         file name: jackson-databind-2.10.2.jar
49         ]]>
50     </notes>
51     <packageUrl regex="true">^pkg:maven/com\.fasterxml\.jackson\.core/jackson\-databind@.*$</packageUrl>
52         <cve>CVE-2020-25649</cve>
53         <cve>CVE-2020-36518</cve>
54         <cve>CVE-2022-42003</cve>
55         <cve>CVE-2022-42004</cve>
56 </suppress>
57 <suppress>
58     <notes>
59         <![CDATA[
60         file name: log4j-api-2.12.1.jar
61         ]]>
62     </notes>
63     <packageUrl regex="true">^pkg:maven/org\.apache\.logging\.log4j/log4j\-api@.*$</packageUrl>
64         <cve>CVE-2020-9488</cve>
65 </suppress>
66 <suppress>
67     <notes>
68         <![CDATA[
69         file name: logback-core-1.2.3.jar
70         ]]>
71     </notes>
72     <packageUrl regex="true">^pkg:maven/ch\.qos\.logback/logback\-core@.*$</packageUrl>
73         <cve>CVE-2021-42550</cve>
74 </suppress>
75 <suppress>
76     <notes>
77         <![CDATA[
78         file name: snakeyaml-1.25.jar

```

```

79     </notes>
80   </packageUrl regex="true">^pkg:maven/org\.yaml/snakeyaml@.*$</packageUrl>
81   <cve>CVE-2022-1471</cve>
82   <cve>CVE-2017-18640</cve>
83   <cve>CVE-2022-25857</cve>
84   <cve>CVE-2022-38749</cve>
85   <cve>CVE-2022-38751</cve>
86   <cve>CVE-2022-38752</cve>
87   <cve>CVE-2022-41854</cve>
88   <cve>CVE-2022-38750</cve>
89 </suppress>
90 <suppress>
91   <notes>
92     <![CDATA[
93       file name: spring-boot-2.2.4.RELEASE.jar
94     ]]>
95   </notes>
96   <packageUrl regex="true">^pkg:maven/org\.springframework\.boot/spring\-boot@.*$</packageUrl>
97   <cve>CVE-2022-27772</cve>
98 </suppress>
99 <suppress>
100   <notes>
101     <![CDATA[
102       file name: spring-boot-starter-web-2.2.4.RELEASE.jar
103     ]]>
104   </notes>
105   <packageUrl regex="true">^pkg:maven/org\.springframework\.boot/spring\-boot\-starter\-web@.*$</packageUrl>
106   <cve>CVE-2022-27772</cve>
107 </suppress>
108 <suppress>
109   <notes>
110     <![CDATA[
111       file name: spring-core-5.2.3.RELEASE.jar
112     ]]>
113   </notes>
114   <packageUrl regex="true">^pkg:maven/org\.springframework/spring\-core@.*$</packageUrl>
115   <cve>CVE-2022-22965</cve>
116   <cve>CVE-2021-22118</cve>
117   <cve>CVE-2020-5421</cve>
118   <cve>CVE-2022-22950</cve>
119

```

```

120         <cve>CVE-2022-22971</cve>
121         <cve>CVE-2023-20861</cve>
122         <cve>CVE-2022-22968</cve>
123         <cve>CVE-2022-22970</cve>
124         <cve>CVE-2021-22060</cve>
125         <cve>CVE-2021-22096</cve>
126     </suppress>
127 <suppress>
128     <notes>
129         <![CDATA[
130         file name: spring-web-5.2.3.RELEASE.jar
131         ]]>
132     </notes>
133     <packageUrl regex="true">^pkg:maven/org\.springframework/spring\-web@.*$</packageUrl>
134         <cve>CVE-2016-100027</cve>
135         <cve>CVE-2022-22965</cve>
136         <cve>CVE-2021-22118</cve>
137         <cve>CVE-2020-5421</cve>
138         <cve>CVE-2022-22950</cve>
139         <cve>CVE-2022-22971</cve>
140         <cve>CVE-2023-20861</cve>
141         <cve>CVE-2022-22968</cve>
142         <cve>CVE-2022-22970</cve>
143         <cve>CVE-2021-22060</cve>
144         <cve>CVE-2021-22096</cve>
145     </suppress>
146 <suppress>
147     <notes>
148         <![CDATA[
149         file name: spring-webmvc-5.2.3.RELEASE.jar
150         ]]>
151     </notes>
152     <packageUrl regex="true">^pkg:maven/org\.springframework/spring\-webmvc@.*$</packageUrl>
153         <cve>CVE-2022-22965</cve>
154         <cve>CVE-2021-22118</cve>
155         <cve>CVE-2020-5421</cve>
156         <cve>CVE-2022-22950</cve>
157         <cve>CVE-2022-22971</cve>
158         <cve>CVE-2023-20861</cve>
159         <cve>CVE-2022-22968</cve>
160         <cve>CVE-2022-22970</cve>
161         <cve>CVE-2021-22060</cve>
162         <cve>CVE-2021-22096</cve>
163     </suppress>
164 <suppress>
165     <notes>
166         <![CDATA[
167         file name: tomcat-embed-core-9.0.30.jar
168         ]]>
169     </notes>
170     <packageUrl regex="true">^pkg:maven/org\.apache\.tomcat\.embed/tomcat\-embed\-core@.*$</packageUrl>
171         <cve>CVE-2020-1938</cve>
172         <cve>CVE-2020-11996</cve>
173         <cve>CVE-2020-13934</cve>
174         <cve>CVE-2020-13935</cve>
175         <cve>CVE-2020-17527</cve>
176         <cve>CVE-2021-25122</cve>
177         <cve>CVE-2021-41079</cve>
178         <cve>CVE-2022-29885</cve>
179         <cve>CVE-2022-42252</cve>
180         <cve>CVE-2020-9484</cve>
181         <cve>CVE-2021-25329</cve>
182         <cve>CVE-2021-30640</cve>
183         <cve>CVE-2022-34305</cve>
184         <cve>CVE-2021-24122</cve>
185         <cve>CVE-2021-33037</cve>
186         <cve>CVE-2019-17569</cve>
187         <cve>CVE-2020-1935</cve>
188         <cve>CVE-2020-13943</cve>
189         <cve>CVE-2023-28708</cve>
190         <cve>CVE-2021-43980</cve>
191     </suppress>
192 <suppress>
193     <notes>
194         <![CDATA[
195         file name: tomcat-embed-websocket-9.0.30.jar
196         ]]>
197     </notes>
198     <packageUrl regex="true">^pkg:maven/org\.apache\.tomcat\.embed/tomcat\-embed\-websocket@.*$</packageUrl>
199         <cve>CVE-2020-1938</cve>
200         <cve>CVE-2020-8022</cve>
201         <cve>CVE-2020-11996</cve>

```

```

202      <cve>CVE-2020-13934</cve>
view Menu  <cve>CVE-2020-13935</cve>
204      <cve>CVE-2020-17527</cve>
205      <cve>CVE-2021-25122</cve>
206      <cve>CVE-2021-41079</cve>
207      <cve>CVE-2022-29885</cve>
208      <cve>CVE-2022-42252</cve>
209      <cve>CVE-2020-9484</cve>
210      <cve>CVE-2021-25329</cve>
211      <cve>CVE-2021-30640</cve>
212      <cve>CVE-2022-34305</cve>
213      <cve>CVE-2021-24122</cve>
214      <cve>CVE-2021-33037</cve>
215      <cve>CVE-2019-17569</cve>
216      <cve>CVE-2020-1935</cve>
217      <cve>CVE-2020-13943</cve>
218      <cve>CVE-2023-28708</cve>
219      <cve>CVE-2021-43980</cve>
220  </suppress>
221  <suppress>
222    <notes>
223      <![CDATA[
224        file name: logback-classic-1.2.3.jar
225      ]]>
226    </notes>
227    <packageUrl regex="true">^pkg:maven/ch\.qos\.logback/logback\-classic@.*$</packageUrl>
228      <cve>CVE-2021-42550</cve>
229  </suppress>
230  <suppress>
231    <notes>
232      <![CDATA[
233        file name: spring-boot-starter-2.2.4.RELEASE.jar
234      ]]>
235    </notes>
236    <packageUrl regex="true">^pkg:maven/org\.springframework\.boot/spring\-boot\-starter@.*$</packageUrl>
237      <cve>CVE-2022-27772</cve>
238  </suppress>
239  <suppress>
240    <notes>
241      <![CDATA[
242        file name: spring-aop-5.2.3.RELEASE.jar
243      ]]>
244    </notes>
245    <packageUrl regex="true">^pkg:maven/org\.springframework/spring\-aop@.*$</packageUrl>
246      <cve>CVE-2022-22965</cve>
247      <cve>CVE-2021-22118</cve>
248      <cve>CVE-2020-5421</cve>
249      <cve>CVE-2022-22950</cve>
250      <cve>CVE-2022-22971</cve>
251      <cve>CVE-2023-20861</cve>
252      <cve>CVE-2022-22968</cve>
253      <cve>CVE-2022-22970</cve>
254      <cve>CVE-2021-22060</cve>
255      <cve>CVE-2021-22096</cve>
256  </suppress>
257  <suppress>
258    <notes>
259      <![CDATA[
260        file name: spring-boot-starter-json-2.2.4.RELEASE.jar
261      ]]>
262    </notes>
263    <packageUrl regex="true">^pkg:maven/org\.springframework\.boot/spring\-boot\-starter\-json@.*$</packageUrl>
264      <cve>CVE-2022-27772</cve>
265  </suppress>
266  <suppress>
267    <notes>
268      <![CDATA[
269        file name: spring-boot-autoconfigure-2.2.4.RELEASE.jar
270      ]]>
271    </notes>
272    <packageUrl regex="true">^pkg:maven/org\.springframework\.boot/spring\-boot\-autoconfigure@.*$</packageUrl>
273      <cve>CVE-2022-27772</cve>
274  </suppress>
275  <suppress>
276      <cve>CVE-2022-27772</cve>
277  </suppress>
278  <suppress>
279    <notes>
280      <![CDATA[
281        file name: spring-boot-starter-logging-2.2.4.RELEASE.jar
282      ]]>

```

```

283     </notes>
284     <packageUrl regex="true">^pkg:maven/org\.springframework\.boot/spring\-boot\-starter\-logging@.*$/packageUrl>
285     <cve>CVE-2022-27772</cve>
286 </suppress>
287 <suppress>
288 <notes>
289     <![CDATA[
290 file name: spring-boot-starter-validation-2.2.4.RELEASE.jar
291 ]]>
292 </notes>
293 <packageUrl regex="true">^pkg:maven/org\.springframework\.boot/spring\-boot\-starter\-validation@.*$/packageUrl>
294 <cve>CVE-2022-27772</cve>
295 </suppress>
296 <suppress>
297 <notes>
298     <![CDATA[
299 file name: spring-jcl-5.2.3.RELEASE.jar
300 ]]>
301 </notes>
302 <packageUrl regex="true">^pkg:maven/org\.springframework/spring\-jcl@.*$/packageUrl>
303 <cve>CVE-2022-22965</cve>
304 <cve>CVE-2021-22118</cve>
305 <cve>CVE-2020-5421</cve>
306 <cve>CVE-2022-22950</cve>
307 <cve>CVE-2022-22971</cve>
308 <cve>CVE-2023-20861</cve>
309 <cve>CVE-2022-22968</cve>
310 <cve>CVE-2022-22970</cve>
311 <cve>CVE-2021-22060</cve>
312 <cve>CVE-2021-22096</cve>
313 </suppress>
314 <suppress>
315 <cve>CVE-2022-22965</cve>
316 <cve>CVE-2021-22118</cve>
317 <cve>CVE-2020-5421</cve>
318 <cve>CVE-2022-22950</cve>
319 <cve>CVE-2022-22971</cve>
320 <cve>CVE-2023-20861</cve>
321 <cve>CVE-2022-22968</cve>
322 <cve>CVE-2022-22970</cve>
323 <cve>CVE-2021-22060</cve>
324 <cve>CVE-2021-22096</cve>
325 </suppress>
326 <suppress>
327 <notes>
328     <![CDATA[
329 file name: spring-context-5.2.3.RELEASE.jar
330 ]]>
331 </notes>
332 <packageUrl regex="true">^pkg:maven/org\.springframework/spring\-context@.*$/packageUrl>
333 <cve>CVE-2022-22965</cve>
334 <cve>CVE-2021-22118</cve>
335 <cve>CVE-2020-5421</cve>
336 <cve>CVE-2022-22950</cve>
337 <cve>CVE-2022-22971</cve>
338 <cve>CVE-2023-20861</cve>
339 <cve>CVE-2022-22968</cve>
340 <cve>CVE-2022-22970</cve>
341 <cve>CVE-2021-22060</cve>
342 <cve>CVE-2021-22096</cve>
343 </suppress>
344 <suppress>
345 <notes>
346     <![CDATA[
347 file name: spring-expression-5.2.3.RELEASE.jar
348 ]]>
349 </notes>
350 <packageUrl regex="true">^pkg:maven/org\.springframework/spring\-expression@.*$/packageUrl>
351 <cve>CVE-2022-22965</cve>
352 <cve>CVE-2021-22118</cve>
353 <cve>CVE-2020-5421</cve>
354 <cve>CVE-2022-22950</cve>
355 <cve>CVE-2022-22971</cve>
356 <cve>CVE-2023-20861</cve>
357 <cve>CVE-2022-22968</cve>
358 <cve>CVE-2022-22970</cve>
359 <cve>CVE-2021-22060</cve>
360 <cve>CVE-2021-22096</cve>
361 </suppress>
362 <suppress>
363 <notes>
364     <![CDATA[
365 file name: json-smart-2.3.jar
366 ]]>
367 </notes>
368 <packageUrl regex="true">^pkg:maven/net\.minidev/json\-smart@.*$/packageUrl>
369 <cve>CVE-2021-31684</cve>
370 <cve>CVE-2023-1370</cve>
371 <cve>CVE-2021-27568</cve>
372 </suppress>
373 </suppressions>

```

Dependency Test



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 tool and the reporting provided constitutes acceptance for use in an AS IS condition, and there are NO warranties, implied or otherwise, 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 arising out of or in connection with the use of this tool, the analysis performed, or the resulting report.

[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*: 8.2.1
- *Report Generated On*: Thu, 13 Apr 2023 13:35:36 -0400
- *Dependencies Scanned*: 49 (28 unique)
- *Vulnerable Dependencies*: 0
- *Vulnerabilities Found*: 0
- *Vulnerabilities Suppressed*: 61
- ...

Summary

Display: [Showing Vulnerable Dependencies \(click to show all\)](#)

Dependency Vulnerability IDs Package Highest Severity CVE Count Confidence Evidence Count

Dependencies

Suppressed Vulnerabilities

This report contains data retrieved from the [National Vulnerability Database](#).
This report may contain data retrieved from the [CISA Known Exploited Vulnerability Catalog](#).
This report may contain data retrieved from the [Github Advisory Database \(via NPM Audit API\)](#).

6. Functional Testing

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

SslServerApplication.java

```
10 import io.github.bucket4j.Bandwidth;
11 import io.github.bucket4j.Bucket;
12 import io.github.bucket4j.Refill;
13
14 import java.time.Duration;
15
16 private final Bucket bucket;
17
18 public ServerController() {
19     Bandwidth limit = Bandwidth.classic(20, Refill.greedy(20, Duration.ofMinutes(1)));
20     this.bucket = Bucket.builder()
21         .addLimit(limit)
22         .build();
23 }
24 // Maps a route to the check sum
25 @RequestMapping(value="/hash")
26 final public String checkSumPage() {
27     if (bucket.tryConsume(1)) {
28         final String data = "Jesse Fjestad";
29         final String checksum = getSHA(data);
30         return "data: " + data + "<br>" + "Checksum hash: " + checksum;
31     }
32     return "Error too many requests";
33 }
34
35
36
37
38
39
40
41
42
43
44
45
46
47
```

Application.properties

```
9 server.error.whitelabel.enabled=false
10 server.error.path=/error
```

Pom.xml

```

42<Ⓜ      <dependency>
43          <groupId>org.springframework.boot</groupId>
44          <artifactId>spring-boot-starter-security</artifactId>
45      </dependency>
46<Ⓜ      <dependency>
47          <groupId>org.springframework</groupId>
48          <artifactId>spring-core</artifactId>
49          <version>5.3.26</version>
50      </dependency>
51<Ⓜ      <dependency>
52          <groupId>org.springframework.security</groupId>
53          <artifactId>spring-security-web</artifactId>
54          <version>5.6.9</version>
55      </dependency>
56<Ⓜ      <dependency>
57          <groupId>org.springframework.security</groupId>
58          <artifactId>spring-security-core</artifactId>
59          <version>5.6.9</version>
60      </dependency>
61<Ⓜ      <dependency>
62          <groupId>org.springframework.security</groupId>
63          <artifactId>spring-security-config</artifactId>
64          <version>5.6.9</version>
65      </dependency>
66<Ⓜ      <dependency>
67<Ⓜ          <groupId>
68              org.springframework.security
69          </groupId>
70          <artifactId>spring-security-crypto</artifactId>
71          <version>5.6.9</version>
72      </dependency>
73<Ⓜ      <dependency>
74          <groupId>org.thymeleaf</groupId>
75          <artifactId>thymeleaf-spring5</artifactId>
76          <version>3.0.13.RELEASE</version>
77  </dependency>
78<Ⓜ      <dependency>
79          <groupId>org.thymeleaf</groupId>
80          <artifactId>thymeleaf</artifactId>
81          <version>3.0.13.RELEASE</version></dependency>
82<Ⓜ      <dependency>
83          <groupId>com.github.vladimir-bukhtoyarov</groupId>
84          <artifactId>bucket4j-core</artifactId>
85          <version>7.6.0</version>
86  </dependency>

```

SecurityConfig.java

```

1  package com.snhu.sslserver;
2
3<Ⓜ import org.springframework.context.annotation.Bean;
15
16 @Configuration
17 @EnableWebSecurity
18 public class SecurityConfig {
19
20<Ⓜ      public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
21          http
22              .authorizeRequests()
23              .antMatchers(HttpMethod.GET, "/login");
24          return http.build();
25      }
26
27<Ⓜ      @Bean
28      public UserDetailsService userDetailsService() {
29          UserDetails user =
30              User.withUsername("user1")
31                  .password(passwordEncoder().encode("user1Pass"))
32                  .roles("USER")
33                  .build();
34          return new InMemoryUserDetailsManager(user);
35      }
36
37<Ⓜ      @Bean
38      public PasswordEncoder passwordEncoder() {
39          return new BCryptPasswordEncoder();
40      }
41  }

```


ErrController.java

```
1 package com.snhu.sslserver;
2
3 import org.springframework.boot.web.servlet.error.ErrorController;
4 import org.springframework.stereotype.Controller;
5 import org.springframework.web.bind.annotation.RequestMapping;
6
7 @Controller
8 public class ErrController implements ErrorController {
9
10     @RequestMapping("/error")
11     public String handleError() {
12         return "error";
13     }
14
15     @Override
16     public String getErrorPath() {
17         return "/error";
18     }
19 }
20
```

Suppression.xml


A false positive was detected in a subsequent dependency check. It is a confirmed false positive because the dependency version used was not in the versions listed in the vulnerability. Therefore, it was added to the suppression.xml.

```
373 <suppress>
374   <notes>
375     <![CDATA[
376       file name: spring-security-crypto-5.6.9.jar
377     ]]>
378   </notes>
379   <packageUrl regex="true">^pkg:maven/org\.springframework\.security/spring\-security\-crypto@.*$</packageUrl>
380   <cve>CVE-2020-5408</cve>
381 </suppress>
```

A new vulnerability was detected in one of the initial dependencies on April 13, 2023.

```
360 -----
361 <cve>CVE-2023-20863</cve>
```

Final Dependency Report

**DEPENDENCY-CHECK**

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 tool and the reporting provided constitutes acceptance for use in an AS IS condition, and there are NO warranties, implied or otherwise, 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 arising out of or in connection with the use of this tool, the analysis performed, or the resulting report.

[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: 8.2.1
- Report Generated On: Fri, 14 Apr 2023 22:05:34 -0400
- Dependencies Scanned: 61 (37 unique)
- Vulnerable Dependencies: 0
- Vulnerabilities Found: 0
- Vulnerabilities Suppressed: 61
- ...

Summary

Display: [Showing Vulnerable Dependencies \(click to show all\)](#)

Dependency	Vulnerability IDs	Package	Highest Severity	CVE Count	Confidence	Evidence Count
------------	-------------------	---------	------------------	-----------	------------	----------------

Dependencies

Suppressed Vulnerabilities

This report contains data retrieved from the [National Vulnerability Database](#).
This report may contain data retrieved from the [CISA Known Exploited Vulnerability Catalog](#).

7. Summary

The initial area of security improved upon was the Cryptography of the application. This was addressing the specific needs and desires of the client Artemis Financial. The first method used to increase Cryptography security was the implementation of a checksum. This was achieved using an instance of the MessageDigest class that hashed the data utilizing the SHA-256 algorithm. After the original data was hashed using the MessageDigest instance, it was converted into a hexadecimal string via the BytesToHex function. This hexadecimal string was the checksum used to validate the data integrity of the given static data. The second method used to increase Cryptographic security was ensuring the website only used HTTPS. To do this, the application.properties file was modified to only allow traffic through port 8443 which requires HTTPS connections, and a certificate was added and implemented. However, because the certificate was self-signed, it is still regarded by browsers as insecure. This would change in production when a certificate from a certificate authority is used. After refactoring this code, a dependency report was conducted which used a suppression.xml file. The suppression.xml file was used to suppress any vulnerabilities in the application before the refactoring process began. It allowed me to analyze only the portions of code that I was refactoring and prevent the inclusion of new vulnerabilities.

The next area of security that was improved upon was the Code Quality of the application. This was strengthened in two ways the implementation of the SecurityConfig.java file created access control and the creation of a rate-limiting system using the bucket4j library. In the SecurityConfig.java file, A SecurityFilterChain method was created to funnel all HTTPS requests to a centralized login page. The userDetailsService method was created to handle the usernames and passwords of users. Currently, the username and password are hard-coded in plain text as “user1” and “user1Pass” respectively. This should be changed during production to

values stored in an encrypted database. However, during development, I did not have access to the correct database and went with the hard-coded values. The `passwordEncoder` method is used to encode the password. Rate limiting was added in the `SslServerApplication.java` file via the `bucket4j` library. The `ServerController` constructor method creates a bucket instance that prevents more than 20 requests per minute. A check is then added to the checksum page to ensure the user has not exceeded the limit. To implement these features, the libraries `spring-boot-starter-security`, `spring-security-web`, `spring-security-core`, `spring-security-config`, `spring-security-crypto`, and `spring-core` were either added or updated to more current versions to fix bugs and resolve vulnerabilities. Additionally, a main page was added to hide system information because certain properties were being displayed in place of a main page when navigating to `https://localhost:8443/`.

The final area improved in the application is Code Error. This was done by creating the `ErrController.java` file and modifying the `application.properties` file. The `ErrController` file creates a controller to handle HTTP errors and deploy a custom web page to the user. The `server.error.whitelabel.enabled` and the `server.error.path` properties were added to prevent the server's generic 404 error page from displaying. In addition, the `thymeleaf` and `thymeleaf-spring5` dependencies were added to enhance resource management which would allow the application to find the appropriate resource.

8. Industry Standard Best Practices

Using a checksum allows the verification of data integrity. This ensures that users can be assured the data they expect is the data they get. It also reduces the risk of malware by preventing files and data that appear similar to the user but would have devastating consequences on the system. This was employed to assure the users that the files given to them are what they

seem. Errors are being handled properly to prevent data from leaking to the user. This ensures the confidentiality of system information and often provides a better user experience. To increase the application's defense against DOS attacks, I used rate limiting which prevents the over-expenditure of system resources by creating resource usage limits within a certain amount of time. This prevents users from crashing the application due to an overuse of the API. Access control was the final industry standard principle implemented. The importance of proper access control cannot be understated. Without it, any user of your site can make changes or access confidential parts of the system. This principle should not only be applied to essential aspects of the application, but it should remain an integral part of the overall security plan. This principle should begin with the initial contact made between the user and the application, which is why I felt the need to include an access control mechanism in the application.

The success of a business is largely dependent on the level of trust it has with its users. Cyber security is an important aspect of maintaining that trust in a digital world. If a breach were to occur, there may be a leak of financial data, user information, or system information, but those specific things can be regained to an extent. The biggest issue is the users' trust because it is the hardest to regain and the quickest to fall. Maintaining a secure system and following industry standards is one of the best ways to ensure the continued trust of your users. Even if the business's systems do not experience a breach, the company could still lose face if they appear susceptible to attacks. This is especially true in the world of social media where the news of a systems breach or system vulnerabilities travels like wildfire. If a business's system does not appear secure, it can greatly harm the trust of its users even if its system is not breached.

References

National Institute of Standards and Technology. (2015, August). *Secure Hash Standard (SHS)*

<https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf>

National Institute of Standards and Technology. (2022, December 15). *NIST Retires SHA-1*

Cryptographic Algorithm. <https://www.nist.gov/news-events/news/2022/12/nist-retires-sha-1-cryptographic-algorithm#:~:text=As%20attacks%20on%20SHA%2D1,31%2C%202030>

Siebert, C. (2021). *Highly Scalable Parallel Checksums*. 2021 IEEE 27th International

Conference on Parallel and Distributed Systems (ICPADS), Parallel and Distributed

Systems (ICPADS), 2021 IEEE 27th International Conference on, ICPADS, 812–818.

<https://doi-org.ezproxy.snhu.edu/10.1109/ICPADS53394.2021.00107>