****

# Practices for Secure Software Report

Table of Contents

[Practices for Secure Software Report 1](#_Toc126608648)

[Document Revision History 3](#_Toc126608649)

[Client 3](#_Toc126608650)

[Instructions 3](#_Toc126608651)

[Developer 4](#_Toc126608652)

[1. Algorithm Cipher 4](#_Toc126608653)

[2. Certificate Generation 4](#_Toc126608654)

[3. Deploy Cipher 4](#_Toc126608655)

[4. Secure Communications 4](#_Toc126608656)

[5. Secondary Testing 5](#_Toc126608657)

[6. Functional Testing 7](#_Toc126608658)

[7. Summary 7](#_Toc126608659)

[8. Industry Standard Best Practices 7](#_Toc126608660)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **02/06/2023** | **Tanner Gaudes** | **Initial version** |

## 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

Tanner Gaudes

## Algorithm Cipher

The encryption algorithm used for the checksum verification is SHA-256, which is an asymmetric encryption algorithm that produces a 256-bit hash from the given data regardless of original size. This checksum verification is used to ensure that data is not altered in transit from the server to the client. If a file that a user uploads to Artemis Financial’s web application is altered while in transit, its checksum value that is recalculated upon arrival will not match the original checksum value. The encryption algorithm’s bit level, 256-bit, is important because the larger the size of the hash output, the less likely it is that a different original value will produce the same hash. The hash function takes data of arbitrary size and converts it to a hashed version of itself that is exactly 256 bits in size via a series of mathematical operations.

Secure random numbers play an important role in software security. These random numbers are not truly random but are instead pseudorandom numbers that are calculated using a mathematical algorithm based on a truly random seed. Regardless, these secure random numbers are often used to create encryption algorithm’s encryption and decryption keys. These keys come in two types: symmetric and asymmetric. Symmetric keys are keys that are the exact same for encryption and decryption and are often used for purposes like certificate verification which does not require as strict security as something like password hashing, for instance. Symmetric keys are inherently less secure than asymmetric keys because if one key is exposed, they are both exposed. On the other hand, asymmetric keys are significantly more secure because they use entirely different keys for encryption and decryption. For this reason, asymmetric keys are almost exclusively used to handle private and sensitive data.

As technology has advanced rapidly, so have security concepts and implementations. The earliest encryption standard, Data Encryption Standard (DES), has been phased out entirely because as computer technology advanced, it was no longer able to resist brute-force attacks and thus became obsolete. This trend has continued over the years, with its successor Triple DES being slowly phased out by the more modern Advanced Encryption Standard (AES) (Arcserve, 2022, pt. 1).

## Certificate Generation

Below are two screenshots of the certificate file export and the file shown in Windows Explorer.

: \Program -export -alias ssprojecttwo -storepass JmSaY4S$ -file projecttwo. cer -keyst 
re proj ecttwokeystore. jks 
ertificate stored in file <projecttwo. cer> prefs.dll 
prołecttwokeystore.iks 
12/24/2022 8:33 PM 
2/5/2023 6:20 PM 
Application exten... 
JKS File 
26 KB 

## Deploy Cipher

The checksum verification screenshot is included as part of the browser screenshot below in part 4.

## Secure Communications

Below is a screenshot of the checksum verification and the secure website verification. I also opened the 'Page Info’ tab for extra certificate details.

04. Module Five- CS-305-J7341 sci X 
DZL Module Five Coding Assignmel X DZL Project Two Guidelines and Ruk X 
O (PA https://localhost:8443/hash 
O 
How to enable HITPS in a Sprin X 
localhost:8443/hash 
Page Info — https://IocaIhost8443/hash 
Twitch 
• YouTube 
Yahoo Mail 
Wowhead TBC 
Hulu 
Netflix Google Drive 
My Account Account.. 
Home - Students 
data: Tanner Gaudes Cipher algorithm : 
SHA-256 Hash value: E13B32BF6F62020F33722FOFBB47C2437ESBDAE969317739E59A7DD65D15DB37 
General Permissions 
Website Identity 
Security 
Website: 
Owner: 
local host 
This website does not supply ownership information. 
Verified by. SNHU 
Privacy & %tory 
Have visited this website prior to today? 
Is this website storing information on my 
computer? 
Have saved any passwords for this website? 
Technical Details 
Yes, 108 times 
Yes, cookies and 45 bytes of 
site data 
No 
View Ce rtficate 
Clear Cookies and Site Data 
View Saved Passwords 
Connection Encrypted (TLS AES_128 GCM SHA2S6, 128 bit keys, TLS 1 3) 
The page you are viewing was encrypted before being transmitted over the Internet. 
Encryption makes it dfficult for unauthorized people to view information traveling between computers. It is 
therefore unlikely that anyone read this page as it traveled across the network. 
Help 

## Secondary Testing

Below are screenshots of the successfully executed install build with the Dependency Check and the Dependency Check HTML output. Note that numerous false positives needed to be suppressed.

Run: 
9 
Di ssl-server [install] 
ssl-server [install]: At 2/5/2023 6:34 PM 
LINFOJ 
Installing C: 
[INFO] 
Installing C: 
[INFO] - 
[INFO] 
BUILD SUCCESS 
[INFO] - 
[INFO] 
Total time: 
[INFO] 
Finished at: 
[INFO] - 
stuff\CS-305 \module 
7\CS 
7\CS 
305 
305 
Project 
project 
TWO 
TWO 
Code 
Code 
Base\ssl 
Base\ssl 
-server 
-server 
13.490 s 
p Version Control 
Run 
TODO 
O Problems 
Terminal 
O Services 
•S Build 
Process finished with exit code 0 
Dependencies DEPENDENCY-CHECK 
is —n tm' *'faming effort of 3rd gositivæ rr.y exist in 
OWASF out of in with of this 
How to read the report I Suppressing false positives I Getting Help: github issues 
Sponsor 
Project: ssl-server 
I-SNAPSHOT 
Scan Information 
• dependency-check version: 7 _4_4 
• Report Generated On: sun, Feb 2023 
• Dependencies Scanned: 49 (42 unique) 
• Vulnerable Dependencies: 0 
• Vulnerabilities Found: 0 
• Vulnerabilities Suppressed: 161 
Summary 
Display: (click to show all) 
Dependency Vulnerability IDs Package Highest Severity 
Dependencies 
Suppressed Vulnerabilities 
CVE Count 
Confidence 
Evidence Count 
This report contains data retrieved from the National 
This report may contain data retrieved from the NPM Public Advisories. 
This report may contain data retrieved from RetiredS_ 
This report may contain data retrieved from the Sonatupe OSS Index. 

## Functional Testing

Below is a screenshot of the REST application running without errors.

Run: SsIServerAppIication 
2023-02-05 
18 
2023-02-05 
18 
2023-02-05 
18 
2023-02-05 
18 
2023-02-05 
18 
2023-02-05 
18 
2023-02-05 
18 
2023-02-05 
18 
• 16.403 
2023-02-05 
18 
2023-02-05 
18 
INFO 
INFO 
INFO 
INFO 
INFO 
INFO 
INFO 
INFO 
INFO 
INFO 
22280 
22280 
22280 
22280 
22280 
22280 
22280 
22280 
22280 
22280 
main J 
main] 
main] 
main] 
main] 
main] 
main] 
main] 
main] 
main] 
com.snhu . sslserver . Ss1ServerApp1ication 
com.snhu . sslserver . Ss1ServerApp1ication 
o. s . b . w. embedded . tomcat . TomcatWebServer 
o. apache . catalina . core . Standardservice 
org.apache.catalina.core.StandardEngine 
o.a.c.c.C. [Tomcat] . [localhost] . C/ ] 
o. s . web . context . ContextLoader 
o. s . s . concurrent . Threadp001TaskExecutor 
o. s . b . w. embedded . tomcat . TomcatWebServer 
snhu . sslserver . Ss1ServerApp1ication 
com. 
Starting Ss1ServerApp1ication on DESKTOP-6RP2TAT with PID 22280 (starte 
. No active profile set, falling back to default profiles: default 
. Tomcat initialized with port(s): 8443 (https) 
Starting service [Tomcat] 
Starting Servlet engine: [Apache Tomcat/9.0.30] 
Initializing Spring embedded WebApp1icationContext 
Root WebApp1icationContext: initialization completed in 1219 ms 
Initializing Executorservice ' applicationTaskExecutor ' 
. Tomcat started on port(s): 8443 (https) with context path 
Started Ss1ServerApp1ication in 2.515 seconds (JVM running for 2.824) 
p Version Control 
Run 
TODO 
O Problems 
Terminal 
O Services 
Build 
Dependencies 

## Summary

Artemis Financial’s web application has been modified to operate securely over the internet. This includes implementing SSL and a checksum verification to ensure that files uploaded to the application were not tampered with while in transit. A static testing tool known as the Dependency Check was also implemented to identify any known vulnerabilities exposed by the application’s dependencies.

The areas of security that were addressed by my modification of the application and their reasoning are as follows:

1. Input Validation (Secure Input and Representations)
   1. Files that users upload to Artemis Financial’s web application can be considered input. This input is validated via the newly added checksum verification to ensure that the files are not tampered with while in transit.
2. Cryptography (Encryption Use and Vulnerabilities)
   1. Cryptographic algorithms are used in both the checksum verification (see above) and the SSL/HTTPS certificates that are used for making a secure connection between the server and the clients.
3. Client/Server (Secure Distributed Composing)
   1. This area of security is relevant in implementing SSL/HTTPS connections.
4. Code Error (Secure Error Handling)
   1. The function which creates a checksum and prints it to the client’s browser may throw a NoSuchAlgorithmException. The function that gets the correct hashing algorithm is now wrapped in a try block that catches the potential exception without crashing the server or revealing sensitive information to clients.
5. Code Quality (Secure Coding Practices / Patterns)
   1. Use of the OWASP Dependency Check and further manual code review ensures that the newly added code is up to industry standards and does not expose any new vulnerabilities.

Implementing multiple layers of security into the application is critical to keeping it error-free and secure; levels of redundancy can mean the difference between an attacker being stopped before causing damage or being able to wreak havoc on an entire system. Industry standard best practices like the use SSL/HTTPS, checksum verification, static testing, and manual code review helped to ensure the application’s security.

## Industry Standard Best Practices

It is standard that websites implement SSL/HTTPS to ensure a secure connection between the client and the server. The encryption algorithms used to encrypt data traveling through this connection should have a minimum bit level of 128-bits to ensure that the encrypted data can resist brute-force attacks. A checksum verification is also used to ensure that files uploaded to the web application by Artemis Financial’s users are not altered while in transit. Additionally, the static testing tool OWASP Dependency Check (referenced below) was used to identify any known vulnerabilities encountered within the application’s code. By implementing multiple layers of security, the application can become significantly more secure and help protect sensitive user data.

Applying industry standards to Artemis Financial’s web application is both required by law and in the best interest of its users. A significant majority of the U.S. population uses digital banking services (Bennett, 2022). These services have access to some of their users’ most private data, including names, addresses, social security numbers, account numbers, and so on. This data must be kept as secure as possible to prevent it from winding up in the wrong hands. As a matter of fact, according to the Federal Trade Commision (FTC) in 2001, customer-facing financial institutions must “protect by encryption all customer information help or transmitted by you both in transit over networks and at rest” (The SSL Store, 2019). Ensuring that this private data remains secure is important in maintaining trust among an application’s users; as end users, we want to keep our private information out of the hands of individuals who want to cause harm.

**References:**

Arcserve. (2022, May 24). 5 common encryption algorithms and the unbreakables of the future. *Arcserve*. Retrieved from: <https://www.arcserve.com/blog/5-common-encryption-algorithms-and-unbreakables-future>

Bennett, R. & Bennett, K. (2022, September 28). Digital banking in 2022: Trends and statistics. *Bankrate*. Retrieved from: <https://www.bankrate.com/banking/digital-banking-trends-and-statistics/>

The SSL Store. (2019, June 28). 10 data privacy and encryption laws every business needs to know. *HashedOut by The SSL Store*. Retrieved from: <https://www.thesslstore.com/blog/10-data-privacy-and-encryption-laws-every-business-needs-to-know/>

OWASP. (n.d.) OWASP Dependency Check. Retrieved from: <https://owasp.org/www-project-dependency-check/>