****

# CS 305 Project One

**Artemis Financial Vulnerability Assessment Report**

Table of Contents

[Document Revision History 3](#_Toc32574607)

[Client 3](#_Toc32574608)

[Instructions 3](#_Toc32574609)

[Developer 4](#_Toc32574610)

[1. Interpreting Client Needs 4](#_Toc32574611)

[2. Areas of Security 4](#_Toc32574612)

[3. Manual Review 4](#_Toc32574613)

[4. Static Testing 4](#_Toc32574614)

[5. Mitigation Plan 4](#_Toc32574615)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **March 14, 2022** | **Kathryn Snow** |  |

## Client



## Instructions

Deliver this completed vulnerability assessment report, identifying your findings of security vulnerabilities and articulating recommendations for next steps to remedy the issues you have found.

Respond to the five steps outlined below and include your findings. Replace the bracketed text on all pages with your own words. If you choose to include images or supporting materials, be sure to insert them throughout.

## Developer

Kathryn Snow

## 1. Interpreting Client Needs

Determine your client’s needs and potential threats and attacks associated with their application and software security requirements. Consider the following regarding how companies protect against external threats based on the scenario information:

* What is the value of secure communications to the company?
* Are there any international transactions that the company produces?
* Are there governmental restrictions about secure communications to consider?
* What external threats might be present now and in the immediate future?
* What are the “modernization” requirements that must be considered, such as the role of open source libraries and evolving web application technologies?

Artemis Financial is an international financial consulting company. Their security needs are crucial to the company’s success. There are many rules and legal expectations to be aware of in their area, like the November 12, 1999, Gramm-Leach-Bliley (GLB) Act which requires companies defined under the law as “financial institutions” to ensure the security and confidentiality of personal information. The Federal trade Commission (FTC) issued the Safeguard’s Rule placing the GLB Act under the FTC’s jurisdiction. The Safeguard’s Rule describes who and how to comply. In July of 2002 the Sarbanes-Oxley Act was passed to provide greater oversight for corporations to deter corporations from committing corporate fraud. And in a document by Congressional Research Service dated March of 2020 entitled “Who Regulates Whom? An Overview of the U.S. Financial Regulatory Framework, states “…U.S. regulators must account for foreign financial firms operating in the United States and foreign regulators must account for U.S. firms operating in their jurisdictions.” This clearly shows there are international restrictions and an expectation of who will oversee them.

Financial service is a highly competitive business. Companies attempt to have the most personal and around the clock services available while handling big data and keeping up with regulations. It easily opens them to potential cybersecurity issues. Financial services make up 35% of all data breaches according to a blog by MindSight. The article lists seven areas most attacked. They are Web Application attacks, DDoS attacks, back-doors and supply-chain attacks, third-fourth-fifth party vendors, global operation risks, emerging technology threats and insider attacks. It was noted that 60% of these attacks are from inside the company with only 25% caused by human error. To handle the modernization of this RESTful API we will use two-step authentication and make certain the libraries currently used are up-to-date.

## 2. Areas of Security

Based on the Vulnerability Assessment Process Flow Diagram Artemis Financial’s software application would benefit from:

* Input Validation – We will review the command input function used in the application and make improvements, if necessary, because it must be secure to prevent injection, the anti-pattern of mixing untrusted data and string-based commands to bypass authentication or gain elevated privileges.
* API (application program interface) – Artemis Financial uses RESTful API. This resource identified by a URL, usually a HTTP URL, connects to preform actions on a resource by using a representation to capture the current or intended state of the resource and transfers that representation. This protects the authentication credentials in transit. We will review and make improvements as necessary.
* Cryptography – We will follow the requirements from the FFIEC (Federal Financial Institution Examination Council). They are clearly outlined in the reference cited.
* Client-Server – The RESTful API has the client-server as the primary connector type. We will review and make improvements in the mutually authenticated client-side certificates to provide protection to our highly privileged web service. The REST service must only provide HTTPS endpoints where access control and JSON Web Tokens (JWT) integrity can be confirmed.
* Code Error – We will review all command input and establish a solid error handling mechanism for input, system calls, database queries and other internal functions.
* Code Quality - We will review existing code and write software that maintains the confidentiality, integrity, and availability of our information by focusing on OWASP’s secure coding practice guidelines.
* Encapsulation – We will review existing code and implement encapsulation, hiding the data object of a class from other classes, reducing the ability of an attack to reach all parts of the application.

## 3. Manual Review

Continue working through the Vulnerability Assessment Process Flow Diagram. Identify all vulnerabilities in the code base by manually inspecting the code.

Input Validation issues are in CRUDController where the input goes directly into a string. Parameterization is necessary protect code. In DocData there is an upcoming *fixme* to protect the code.

## 4. Static Testing

Run a dependency check on Artemis Financial’s software application to identify all security vulnerabilities in the code. Record the output from dependency check report. Include the following:

1. The names or vulnerability codes of the known vulnerabilities
2. A brief description and recommended solutions provided by the dependency check report
3. Attribution (if any) that documents how this vulnerability has been identified or documented previously

**bcprov-jdk15on-1.46.jar**

The Bouncy Castle Crypto package is a Java implementation of cryptographic algorithms. This jar contains JCE provider and lightweight API for the Bouncy Castle Cryptography APIs for JDK 1.5 to JDK 1.7.

Common vulnerabilities and exposures: CVE-2013-1624, CVE-2016-6644, CVE-2015-7940, CVE-2016-1000338, CVE-2016-1000339, CVE-2016-1000341, CVE-2016-1000342, CVE-2016-1000343, CVE-2016-1000344, CVE-2016-1000345, CVE-2016-1000346, CVE-2016-1000352, CVE-2017-13098, CVE-2018-1000613, CVE-2018-5382, CVE-2020-15522, and CVE-2020-26939.

**hibernate-validator-6.0.18.Final.jar**

Hibernate's Bean Validation (JSR-380) reference implementation

Common vulnerabilities and exposures: CVE-2020-10693.

**jackson-databind-2.10.2.jar**

General data-binding functionality for Jackson: works on core streaming API

Common vulnerabilities and exposures: CVE-2020-25649.

**log4j-api-2.12.1.jar**

The Apache Log4j API

Common vulnerabilities and exposures: CVE-2020-9488.

**logback-core-1.2.3.jar**

logback-core module

Common vulnerabilities and exposures: CVE-2021-42550.

**snakeyaml-1.25.jar**

YAML 1.1 parser and emitter for Java

Common vulnerabilities and exposures: CVE-2017-18640.

**spring-aop-5.2.3.RELEASE.jar**

Spring AOP

Common vulnerabilities and exposures: CVE-2020-5421, CVE-2021-22060, CVE-2021-22096, and CVE-2021-22118.

**tomcat-embed-core-9.0.30.jar**

Core Tomcat implementation

Common vulnerabilities and exposures: CVE-2019-17569, CVE-2020-11996, CVE-2020-13934, CVE-2020-13936, CVE-2020-13943, CVE-2020-17527, CVE-2020-1935, CVE-2020-1938, CVE-2020-9484, CVE-2021-24122, CVE-2021-25122, CVE-2021-25329, CVE-2021-30640, CVE-2021-33037, CVE-2021-41079, and CVE-2021-42340.

**tomcat-embed-websocket-9.0.30.jar**

Core Tomcat implementation

Common vulnerabilities and exposures: CVE-2019-17569, CVE-2020-11996, CVE-2020-13934, CVE-2020-13935, CVE-2020-13943, CVE-2020-17527, CVE-2020-1935, CVE-2020-1938, CVE-2020-8022, CVE-2020-9484, CVE-2021-24122, CVE-2021-25122, CVE-2021-25329, CVE-2021-30640, CVE-2021-33037, CVE-2021-41079, and CVE-2021-42340.

## 5. Mitigation Plan

After interpreting your results from the manual review and static testing, identify the steps to remedy the identified security vulnerabilities for Artemis Financial’s software application.

We will update Spring Framework and Apache Tomcat. We will review and implement fixes for code quality to block security threats per OWASP’s Secure Coding Practices guide.

We will run the static testing again and review recommendations offered to determine a method to remedy the security vulnerability. We can compare this report to the new one for the repetitive CVEs and note new ones in the upgrades. Our concern list now is as follows:

CVE-2018-5382-The default BKS keystore use an HMAC that is only 16 bits long, which can allow an attacker to compromise the integrity of a BKS keystore.

CVE-2020-25649-A flaw was found in FasterXML Jackson Databind, where it did not have entity expansion secured properly. This flaw allows vulnerability to XML external entity (XXE) attacks. The highest threat from this vulnerability is data integrity.

CVE-2020-9488-Improper validation of certificate with host mismatch in Apache Log4j SMTP appender. This could allow an SMTPS connection to be intercepted by a man-in-the-middle attack which could leak any log messages sent through that appender.

CVE-2020-5421-In Spring Framework, the protections against RFD attacks from CVE-2015-5211 may be bypassed depending on the browser used through the use of a jsessionid path parameter.

CVE-2021-22060-In Spring Framework, it is possible for a user to provide malicious input to cause the insertion of additional log entries.

CVE-2020-13934- An h2c direct connection error to may occur if a sufficient number of such requests were made, an OutOfMemoryException could occur leading to a denial of service.

CVE-2020-11996-A specially crafted sequence of HTTP/2 requests sent to Apache Tomcat could trigger high CPU usage for several seconds. If a sufficient number of such requests were made on concurrent HTTP/2 connections, the server could become unresponsive.

CVE-2020-1935-The HTTP header parsing code used an approach to end-of-line parsing that allowed some invalid HTTP headers to be parsed as valid. This led to a possibility of HTTP Request Smuggling if Tomcat was located behind a reverse proxy that incorrectly handled the invalid Transfer-Encoding header in a particular manner. Such a reverse proxy is considered unlikely.

CVE-2021-25122-When responding to new h2c connection requests duplicate request headers and a limited amount of request body from one request to another meaning user A and user B could both see the results of user A's request.

CVE-2021-30640-A vulnerability in the JNDI Realm of Apache Tomcat allows an attacker to authenticate using variations of a valid user name and/or to bypass some of the protection provided by the LockOut Realm.

CVE-2021-33037-Tomcat incorrectly ignored the transfer encoding header if the client declared it would only accept an HTTP/1.0 response; - Tomcat honoured the identify encoding; and - Tomcat did not ensure that, if present, the chunked encoding was the final encoding.

CVE-2021-41079-When Tomcat was configured to use NIO+OpenSSL or NIO2+OpenSSL for TLS, a specially crafted packet could be used to trigger an infinite loop resulting in a denial of service.

*Financial Institutions and Customer Information: Complying with the Safeguards Rule*. (2020, July 16). Federal Trade Commission. Retrieved March 19, 2022, from https://www.ftc.gov/business-guidance/resources/financial-institutions-customer-information-complying-safeguards-rule

Lutkevich, B. (2020, December 11). *Sarbanes-Oxley Act*. SearchCIO. Retrieved March 19, 2022, from https://www.techtarget.com/searchcio/definition/Sarbanes-Oxley-Act

Congressional Research Service. (2020, March). *Who Regulates Whom? An Overview of the U.S. Financial Regulatory Framework*. https://sgp.fas.org/crs/misc/R44918.pdf

Climer, S., & Khan, M. (2020, June 11). *Top 7 Cybersecurity Threats To Financial Services: A Cybersecurity Report*. Mindsight. Retrieved March 19, 2022, from https://gomindsight.com/insights/blog/7-cyber-security-threats-to-financial-services/

Ross, B. (2021, October 10). *Financial Services Industry Challenges & Opportunities In 2022 from*. Linchpin SEO. Retrieved March 19, 2022, from https://linchpinseo.com/common-challenges-in-the-financial-services-industry/

Federal Financial Institutions Examination Council. (2021, August). *Authentication and Access to Financial Institution Services and Systems*. ffiec.gov. https://www.ffiec.gov/press/pdf/Authentication-and-Access-to-Financial-Institution-Services-and-Systems.pdf