

**Artemis Financial Vulnerability Assessment Report**

Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Comments** |
| **1.0** | **09/16/2023** | **Timothy Merriman** |  |

Client



Instructions

Submit this completed vulnerability assessment report. Replace the bracketed text with the relevant information. In the report, identify your findings of security vulnerabilities and provide recommendations for the next steps to remedy the issues you have found.

* Respond to the five 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 One Guidelines and Rubric for more detailed instructions about each section of the template.

Developer

Timothy Merriman

* Interpreting Client Needs

Artemis Financial, a financial firm, deals with both internal and external priority information for clients, making secure communication paramount. They handle sensitive financial and client data, including biometrics, social security numbers, and account information, necessitating robust security measures.

To safeguard customer information and trade secrets, information must be masked during storage and transmission. Additionally, keeping software libraries up-to-date is crucial for staying protected against evolving cybersecurity threats.

* Areas of Security

Artemis Financial's security assessment reveals several potential vulnerabilities:

Input Validation: Ensure rigorous input validation to prevent failures and SQL injection.

APIs: Develop a secure API for both internal and external use, defining acceptable data access methods to avoid unauthorized access.

Cryptography: Implement strong encryption for international transfers, complying with regulations in North America and destination countries.

Code Error Handling: Proper error handling is crucial, especially for input validation, to prevent unauthorized access.

Code Quality: Maintain high-quality code to prevent unintentional data exposure and restrict access based on user authorization levels.

* Manual Review

I began the vulnerability assessment by checking for input validation in the POM.XML file and the greeting controller. Input validation wasn't evident in the controller, and I couldn't confirm it due to a lack of output. I also couldn't find a working API, making it challenging for users to interact with the program. The program accepts input via the URL, potentially leading to data leaks in the browser history. There's no data encryption, a crucial concern for international transactions. Error handling was inadequate in the DocData.java class, with only basic try-catch blocks.

While the code quality is good, the absence of a functional API and issues with input validation and data handling in the URL make the program user-unfriendly and raise security concerns.

* Static Testing

log4j-api-2.12.1.jar

cpe:2.3:a:apache:log4j:2.12.1:\*:\*:\*:\*:\*:\*:\*

In Apache Log4j SMTP appender, a host mismatch during certificate validation could enable a man-in-the-middle attack, potentially exposing logged messages sent via SMTPS.  
Upgrade to 2.13.2 for built-in support or set mail.smtp.ssl.checkserveridentity to true in earlier versions for global SMTPS hostname verification.

tomcat-embed-core-9.0.30.jar

cpe:2.3:a:apache:tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*

cpe:2.3:a:apache\_tomcat:apache\_tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*

In Apache Tomcat versions 10.0.0-M1 to 10.0.6, 9.0.0.M1 to 9.0.46, and 8.5.0 to 8.5.66, there was a parsing issue with the HTTP transfer-encoding request header. This flaw could potentially lead to request smuggling when used alongside a reverse proxy.

Upgrade to Apache Tomcat 10.0.6 or later

tomcat-embed-websocket-9.0.30.jar

cpe:2.3:a:apache:tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*

cpe:2.3:a:apache\_software\_foundation:tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*

cpe:2.3:a:apache\_tomcat:apache\_tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*

Apache Tomcat versions 10.0.0-M1 to 10.0.6, 9.0.0.M1 to 9.0.46, and 8.5.0 to 8.5.66 had issues parsing the HTTP transfer-encoding request header, which could potentially lead to request smuggling when used with a reverse proxy.

Upgrade to Apache Tomcat 10.0.6 or later

bcprov-jdk15on-1.46.jar

cpe:2.3:a:bouncycastle:bouncy-castle-crypto-package:1.46:\*:\*:\*:\*:\*:\*:\*

cpe:2.3:a:bouncycastle:bouncy\_castle\_crypto\_package:1.46:\*:\*:\*:\*:\*:\*:\*

cpe:2.3:a:bouncycastle:legion-of-the-bouncy-castle-java-crytography-api:1.46:\*:\*:\*:\*:\*:\*:\*

cpe:2.3:a:bouncycastle:the\_bouncy\_castle\_crypto\_package\_for\_java:1.46:\*:\*:\*:\*:\*:\*:\*

In versions 1.58 to just before 1.60 of the Legion of the Bouncy Castle Java Cryptography APIs, there is a security flaw in the XMSS/XMSS^MT private key deserialization. This vulnerability allows for the execution of unexpected code during deserialization. An attacker could craft a private key containing references to unexpected classes from the classpath, which the application would pick up. This issue was resolved in version 1.60 and later.

Update Bouncycastle to Version 1.60

jackson-databind-2.10.2.jar

cpe:2.3:a:fasterxml:jackson-databind:2.10.2:\*:\*:\*:\*:\*:\*:\*

A vulnerability in FasterXML Jackson Databind could lead to XML external entity (XXE) attacks, primarily posing a data integrity risk.

Update to current version

spring-aop-5.2.3.RELEASE.jar

cpe:2.3:a:pivotal\_software:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*

cpe:2.3:a:springsource:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*

cpe:2.3:a:vmware:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*  
cpe:2.3:a:vmware:springsource\_spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*

In Spring Framework versions 5.2.0 - 5.2.8, 5.1.0 - 5.1.17, 5.0.0 - 5.0.18, 4.3.0 - 4.3.28, and older unsupported versions, 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.

Update to current version

hibernate-validator-6.0.18.Final.jar

cpe:2.3:a:redhat:hibernate\_validator:6.0.18:\*:\*:\*:\*:\*:\*:\*  
In Hibernate Validator version 6.1.2.Final, there's a flaw that allows invalid EL expressions to be evaluated as if they were valid. Attackers can exploit this to bypass input sanitation measures applied to user-controlled data in error messages.  
Upgrade to hibernate-validator-6.0.20

snakeyaml-1.25.jar  
cpe:2.3:a:snakeyaml\_project:snakeyaml:1.25:\*:\*:\*:\*:\*:\*:\*  
In SnakeYAML 1.18, the Alias feature permits entity expansion during loading, resembling CVE-2003-1564, reported on December 11, 2019.  
Migrate to SnakeYAML Engine. It has a configuration option to restrict aliases for collections.

* Mitigation Plan

Updating to the latest versions can mitigate most of these vulnerabilities. Additionally, enhancing security can be achieved by modifying the Snakeyaml version and restricting aliases for collections.