****

# 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** | **03/13/2013** | **Emily Wood** | **Initial Notes** |

## Client



## Developer

Emily Wood

## Interpreting Client Needs

Artemis Financial wants to modernize their operations, which includes developing individual financial plans for their customers. We are implementing an expressive command input function for a complex web application. We are currently using Version 2.6.5 of the spring-data-rest-webmvc in the Spring framework, and we want to use the Spring Expression Language to accomplish the task. Artemis wishes to have their current RESTful API examined for security vulnerabilities.  
  
Secure communication is necessary to deter hackers, ensure privacy of the company and consumers, be compliant with any laws regarding protected information, protect against cyber-attacks, enhance trust of the system, and maintain accountability by ensuring safe logging.

It is unknown at this time if Artemis will be completing any international transactions, but if they want to do that in the future, there are some things to consider. Understanding local laws and regulations are important.  
  
External threats are always something to consider. Since COVID, digital transactions have increased and therefore bringing more opportunities for hackers. Examples include phishing, ransomware, SQL injections, distributed denial-of-service (DDoS) attacks, supply chain attacks, and bank drops (Kost, 2023).

One request of Artemis is to modernize their operations. Open-source software has gained popularity with about 93% of organizations using it (Kirilov, 2020). Open-source software is beneficial in that it offers customization to fit specific needs. However, vulnerabilities are widely distributed. Identifying vulnerabilities early is key to mitigating risk.  
  
Web applications have increased in popularity since more and more people want 24/7 access to data. A study was conducted that showed that attempted cyber-attacks had increased 358% in 2020. Helpful ways to combat this include strong passwords and authentications, multi-factor authentication, real-time protection including on-access scanning, back-u and preventative security measures, and web application firewalls (Ibrahim, 2021).

## Areas of Security

• Input Validation - First, we need to ensure input validation is performed. This is to prevent malfunctions of various downstream components (OWASP Cheat Sheet Series, n.d.).

• APIs - Since we are using a web application, secure API interactions are essential. Since APIs are very commonly used and they enable access to sensitive software functions and data, they are becoming a primary target for attackers (Bright Security, 2022).

• Cryptography - Cryptography should be used for data to ensure attackers cannot make use of it (Bright Security, 2022).

• Client/Server - Client/Server is necessary because we are using an API. Correct certificates are necessary for any data transfers.

• Code Quality - Code Quality is essential because basic secure coding practices and patterns impact how the program behaves. Static testing can also be used to identify areas where code quality is lacking.

• Encapsulation - Encapsulation is important to ensure that boundaries are maintained, and attackers cannot gain unauthorized access.

## Manual Review

CRUD.java  
Text

Description automatically generated

* Naming should be more specific instead of content1 and content2 to allow for readability and maintenance.

CRUDController.java  
Text

Description automatically generated

* There is no authorization in CRUDController.java and can be called by anyone

customer.java  
  
Text

Description automatically generated

* account\_balance is not listed as a private variable and may be visible outside of this class.
* Class should be private
* There is no authentication or input verification within any classes.

## Static Testing

Graphical user interface, text, application, email

Description automatically generated

* After running a dependency check, there were 13 vulnerable dependencies found with 101 vulnerabilities.

|  |  |  |  |
| --- | --- | --- | --- |
| Dependency | Severity | Vulnerability | Description |
| bcprov-jdk15on-1.46.jar | High | CVE-2016-1000338 | It is possible to inject extra elements in the sequence making up the signature and still have it validate, which in some cases may allow the introduction of 'invisible' |
| CVE-2016-1000342 |
| CVE-2016-1000343 | In the Bouncy Castle JCE Provider version 1.55 and earlier the DSA key pair generator generates a weak private key if used with default values. |
| CVE-2016-1000344 | In the Bouncy Castle JCE Provider version 1.55 and earlier the DHIES implementation allowed the use of ECB mode. This mode is regarded as unsafe and support for it has been removed from the provider. |
| CVE-2016-1000352 | In the Bouncy Castle JCE Provider version 1.55 and earlier the ECIES implementation allowed the use of ECB mode. This mode is regarded as unsafe and support for it has been removed from the provider. |
| CVE-2016-1000341 | In the Bouncy Castle JCE Provider version 1.55 and earlier DSA signature generation is vulnerable to timing attack. Where timings can be closely observed for the generation of signatures, the lack of blinding in 1.55, or earlier, may allow an attacker to gain information about the signature's k value and ultimately the private value as well |
| CVE-2016-1000345 | In the Bouncy Castle JCE Provider version 1.55 and earlier the DHIES/ECIES CBC mode vulnerable to padding oracle attack. For BC 1.55 and older, in an environment where timings can be easily observed, it is possible with enough observations to identify when the decryption is failing due to padding. |
| CVE-2017-13098 | BouncyCastle TLS prior to version 1.0.3, when configured to use the JCE (Java Cryptography Extension) for cryptographic functions, provides a weak Bleichenbacher oracle when any TLS cipher suite using RSA key exchange is negotiated. An attacker can recover the private key from a vulnerable application. This vulnerability is referred to as "ROBOT." |
| CVE-2020-15522 | Bouncy Castle BC Java before 1.66, BC C# .NET before 1.8.7, BC-FJA before 1.0.1.2, 1.0.2.1, and BC-FNA before 1.0.1.1 have a timing issue within the EC math library that can expose information about the private key when an attacker is able to observe timing information for the generation of multiple deterministic ECDSA signatures. |
| CVE-2016-1000339 | In the Bouncy Castle JCE Provider version 1.55 and earlier the primary engine class used for AES was AESFastEngine. Due to the highly table driven approach used in the algorithm it turns out that if the data channel on the CPU can be monitored the lookup table accesses are sufficient to leak information on the AES key being used. There was also a leak in AESEngine although it was substantially less. AESEngine has been modified to remove any signs of leakage (testing carried out on Intel X86-64) and is now the primary AES class for the BC JCE provider from 1.56. Use of AESFastEngine is now only recommended where otherwise deemed appropriate. |
| CVE-2015-7940 | The Bouncy Castle Java library before 1.51 does not validate a point is withing the elliptic curve, which makes it easier for remote attackers to obtain private keys via a series of crafted elliptic curve Diffie Hellman (ECDH) key exchanges, aka an "invalid curve attack." |
| 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-2013-1624 | The TLS implementation in the Bouncy Castle Java library before 1.48 and C# library before 1.8 does not properly consider timing side-channel attacks on a noncompliant MAC check operation during the processing of malformed CBC padding, which allows remote attackers to conduct distinguishing attacks and plaintext-recovery attacks via statistical analysis of timing data for crafted packets, a related issue to CVE-2013-0169. |
| CVE-2016-1000346 | In the Bouncy Castle JCE Provider version 1.55 and earlier the other party DH public key is not fully validated. This can cause issues as invalid keys can be used to reveal details about the other party's private key where static Diffie-Hellman is in use. As of release 1.56 the key parameters are checked on agreement calculation. |
| hibernate-validator-6.0.18.Final.jar | Medium | CVE-2020-10693 | A flaw was found in Hibernate Validator version 6.1.2.Final. A bug in the message interpolation processor enables invalid EL expressions to be evaluated as if they were valid. This flaw allows attackers to bypass input sanitation (escaping, stripping) controls that developers may have put in place when handling user-controlled data in error messages. |
| jackson-databind-2.10.2.jar | High | 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-2020-36518 | jackson-databind before 2.13.0 allows a Java StackOverflow exception and denial of service via a large depth of nested objects. |
| CVE-2022-42003 | In FasterXML jackson-databind before 2.14.0-rc1, resource exhaustion can occur because of a lack of a check in primitive value deserializers to avoid deep wrapper array nesting, when the UNWRAP\_SINGLE\_VALUE\_ARRAYS feature is enabled. |
| CVE-2022-42004 | In FasterXML jackson-databind before 2.13.4, resource exhaustion can occur because of a lack of a check in BeanDeserializer.\_deserializeFromArray to prevent use of deeply nested arrays. An application is vulnerable only with certain customized choices for deserialization. |
| log4j-api-2.12.1.jar | Low | 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. |
| logback-core-1.2.3.jar | Medium | CVE-2021-42550 | In logback version 1.2.7 and prior versions, an attacker with the required privileges to edit configurations files could craft a malicious configuration allowing to execute arbitrary code loaded from LDAP servers. |
| snakeyaml-1.25.jar | High | CVE-2017-18640 | The Alias feature in SnakeYAML before 1.26 allows entity expansion during a load operation |
| CVE-2022-25857 | The package org.yaml:snakeyaml from 0 and before 1.31 are vulnerable to Denial of Service (DoS) due missing to nested depth limitation for collections. |
| CVE-2022-38749 | Using snakeYAML to parse untrusted YAML files may be vulnerable to Denial of Service attacks (DOS). If the parser is running on user supplied input, an attacker may supply content that causes the parser to crash by stackoverflow. |
| CVE-2022-38751 |
| CVE-2022-38752 |
| CVE-2022-41854 |
| CVE-2022-38750 |
| spring-boot-2.2.4.RELEASE.jar | High | CVE-2022-27772 | spring-boot versions prior to version v2.2.11.RELEASE was vulnerable to temporary directory hijacking. This vulnerability impacted the org.springframework.boot.web.server. AbstractConfigurableWebServerFactory. createTempDir method. |
| spring-boot-starter-web-2.2.4.RELEASE.jar | High |
| spring-core-5.2.3.RELEASE.jar | Critical | CVE-2022-22965 | A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. |
| CVE-2021-22118 | In Spring Framework, versions 5.2.x prior to 5.2.15 and versions 5.3.x prior to 5.3.7, a WebFlux application is vulnerable to a privilege escalation: by (re)creating the temporary storage directory, a locally authenticated malicious user can read or modify files that have been uploaded to the WebFlux application, or overwrite arbitrary files with multipart request data. |
| CVE-2020-5421 | 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. |
| CVE-2022-22950 | In Spring Framework versions 5.3.0 - 5.3.16 and older unsupported versions, it is possible for a user to provide a specially crafted SpEL expression that may cause a denial of service condition. |
| CVE-2022-22971 | In spring framework versions prior to 5.3.20+ , 5.2.22+ and old unsupported versions, application with a STOMP over WebSocket endpoint is vulnerable to a denial of service attack by an authenticated user. |
| CVE-2022-22968 | In Spring Framework versions 5.3.0 - 5.3.18, 5.2.0 - 5.2.20, and older unsupported versions, the patterns for disallowedFields on a DataBinder are case sensitive which means a field is not effectively protected unless it is listed with both upper and lower case for the first character of the field, including upper and lower case for the first character of all nested fields within the property path. |
| CVE-2022-22970 | In spring framework versions prior to 5.3.20+ , 5.2.22+ and old unsupported versions, applications that handle file uploads are vulnerable to DoS attack if they rely on data binding to set a MultipartFile or javax.servlet.Part to a field in a model object. |
| CVE-2021-22060 | In Spring Framework versions 5.3.0 - 5.3.13, 5.2.0 - 5.2.18, and older unsupported versions, it is possible for a user to provide malicious input to cause the insertion of additional log entries. |
| CVE-2021-22096 | In Spring Framework versions 5.3.0 - 5.3.10, 5.2.0 - 5.2.17, and older unsupported versions, it is possible for a user to provide malicious input to cause the insertion of additional log entries. |
| spring-webmvc-5.2.3.RELEASE.jar | Critical | CVE-2022-22965 | \*Mentioned Previously\* |
| CVE-2021-22118 | \*Mentioned Previously\* |
| CVE-2020-5421 | \*Mentioned Previously\* |
| CVE-2022-22950 | \*Mentioned Previously\* |
| CVE-2022-22971 | \*Mentioned Previously\* |
| CVE-2022-22968 | \*Mentioned Previously\* |
| CVE-2022-22970 | \*Mentioned Previously\* |
| CVE-2021-22060 | \*Mentioned Previously\* |
| CVE-2021-22096 | \*Mentioned Previously\* |
| tomcat-embed-core-9.0.30.jar | Critical | CVE-2020-1938 | When using the Apache JServ Protocol (AJP), care must be taken when trusting incoming connections to Apache Tomcat. Tomcat treats AJP connections as having higher trust than, for example, a similar HTTP connection. |
| CVE-2020-11996 | A specially crafted sequence of HTTP/2 requests sent to Apache Tomcat 10.0.0-M1 to 10.0.0-M5, 9.0.0.M1 to 9.0.35 and 8.5.0 to 8.5.55 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-2020-13934 | An h2c direct connection to Apache Tomcat 10.0.0-M1 to 10.0.0-M6, 9.0.0.M5 to 9.0.36 and 8.5.1 to 8.5.56 did not release the HTTP/1.1 processor after the upgrade to HTTP/2. If a sufficient number of such requests were made, an OutOfMemoryException could occur leading to a denial of service. |
| CVE-2020-13935 | The payload length in a WebSocket frame was not correctly validated in Apache Tomcat 10.0.0-M1 to 10.0.0-M6, 9.0.0.M1 to 9.0.36, 8.5.0 to 8.5.56 and 7.0.27 to 7.0.104. Invalid payload lengths could trigger an infinite loop. Multiple requests with invalid payload lengths could lead to a denial of service. |
| CVE-2020-17527 | Apache Tomcat 10.0.0-M1 to 10.0.0-M9, 9.0.0-M1 to 9.0.39 and 8.5.0 to 8.5.59 could re-use an HTTP request header value from the previous stream received on an HTTP/2 connection for the request associated with the subsequent stream. While this would most likely lead to an error and the closure of the HTTP/2 connection, it is possible that information could leak between requests. |
| CVE-2021-25122 | Apache Tomcat versions 10.0.0-M1 to 10.0.0, 9.0.0.M1 to 9.0.41 and 8.5.0 to 8.5.61 could 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-2021-41079 | Apache Tomcat 8.5.0 to 8.5.63, 9.0.0-M1 to 9.0.43 and 10.0.0-M1 to 10.0.2 did not properly validate incoming TLS packets. 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. |
| CVE-2022-29885 | The documentation of Apache Tomcat 10.1.0-M1 to 10.1.0-M14, 10.0.0-M1 to 10.0.20, 9.0.13 to 9.0.62 and 8.5.38 to 8.5.78 for the EncryptInterceptor incorrectly stated it enabled Tomcat clustering to run over an untrusted network. This was not correct. |
| CVE-2022-42252 | If Apache Tomcat 8.5.0 to 8.5.82, 9.0.0-M1 to 9.0.67, 10.0.0-M1 to 10.0.26 or 10.1.0-M1 to 10.1.0 was configured to ignore invalid HTTP headers via setting rejectIllegalHeader to false (the default for 8.5.x only), Tomcat did not reject a request containing an invalid Content-Length header making a request smuggling attack possible if Tomcat was located behind a reverse proxy that also failed to reject the request with the invalid header. |
| CVE-2020-9484 | When using Apache Tomcat versions 10.0.0-M1 to 10.0.0-M4, 9.0.0.M1 to 9.0.34, 8.5.0 to 8.5.54 and 7.0.0 to 7.0.103 if a) an attacker is able to control the contents and name of a file on the server |
| 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. This issue affects Apache Tomcat 10.0.0-M1 to 10.0.5; 9.0.0.M1 to 9.0.45; 8.5.0 to 8.5.65. |
| CVE-2022-34305 | In Apache Tomcat 10.1.0-M1 to 10.1.0-M16, 10.0.0-M1 to 10.0.22, 9.0.30 to 9.0.64 and 8.5.50 to 8.5.81 the Form authentication example in the examples web application displayed user provided data without filtering, exposing a XSS vulnerability. |
| CVE-2021-24122 | Apache Tomcat versions 10.0.0-M1 to 10.0.0-M9, 9.0.0.M1 to 9.0.39, 8.5.0 to 8.5.59 and 7.0.0 to 7.0.106 were susceptible to JSP source code disclosure in some configurations. |
| CVE-2021-33037 | Apache Tomcat 10.0.0-M1 to 10.0.6, 9.0.0.M1 to 9.0.46 and 8.5.0 to 8.5.66 did not correctly parse the HTTP transfer-encoding request header in some circumstances leading to the possibility to request smuggling when used with a reverse proxy. |
| CVE-2019-17569 | The refactoring present in Apache Tomcat 9.0.28 to 9.0.30, 8.5.48 to 8.5.50 and 7.0.98 to 7.0.99 introduced a regression. The result of the regression was that invalid Transfer-Encoding headers were incorrectly processed leading 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. |
| CVE-2020-1935 | In Apache Tomcat 9.0.0.M1 to 9.0.30, 8.5.0 to 8.5.50 and 7.0.0 to 7.0.99 the HTTP header parsing code used an approach to end-of-line parsing that allowed some invalid HTTP headers to be parsed as valid. |
| CVE-2020-13943 | If an HTTP/2 client connecting to Apache Tomcat 10.0.0-M1 to 10.0.0-M7, 9.0.0.M1 to 9.0.37 or 8.5.0 to 8.5.57 exceeded the agreed maximum number of concurrent streams for a connection (in violation of the HTTP/2 protocol), it was possible that a subsequent request made on that connection could contain HTTP headers - including HTTP/2 pseudo headers - from a previous request rather than the intended headers. |
| CVE-2021-43980 | Apache Tomcat 10.1.0 to 10.1.0-M12, 10.0.0-M1 to 10.0.18, 9.0.0-M1 to 9.0.60 and 8.5.0 to 8.5.77 that could cause client connections to share an Http11Processor instance resulting in responses, or part responses, to be received by the wrong client. |
| tomcat-embed-websocket-9.0.30.jar | Critical | CVE-2020-1938 | \*Mentioned Previously\* |
| CVE-2020-8022 | \*Mentioned Previously\* |
| CVE-2020-11996 | \*Mentioned Previously\* |
| CVE-2020-13934 | \*Mentioned Previously\* |
| CVE-2020-13935 | \*Mentioned Previously\* |
| CVE-2020-17527 | \*Mentioned Previously\* |
| CVE-2021-25122 | \*Mentioned Previously\* |
| CVE-2021-41079 | \*Mentioned Previously\* |
| CVE-2022-29885 | \*Mentioned Previously\* |
| CVE-2022-42252 | \*Mentioned Previously\* |
| CVE-2020-9484 | \*Mentioned Previously\* |
| CVE-2021-25329 | \*Mentioned Previously\* |
| CVE-2021-30640 | \*Mentioned Previously\* |
| CVE-2022-34305 | \*Mentioned Previously\* |
| CVE-2021-24122 | \*Mentioned Previously\* |
| CVE-2021-33037 | \*Mentioned Previously\* |
| CVE-2019-17569 | \*Mentioned Previously\* |
| CVE-2020-1935 | \*Mentioned Previously\* |
| CVE-2020-13943 | \*Mentioned Previously\* |
| CVE-2021-43980 | \*Mentioned Previously\* |

## Mitigation Plan

* Include code to validate any input that is passed through, ie. “name” and “id”.
* Include Try/Catch functions with basic output as to not reveal the nature of any exception.
* Ensure each dependency is updated with the latest version.
* Ensure proper naming conventions with variables to keep code easy to maintain.

**References**:

Kost, E. (2023, January 5). *The 6 biggest cyber threats for financial services in 2023: Upguard*. UpGuard. Retrieved March 13, 2023, from https://www.upguard.com/blog/biggest-cyber-threats-for-financial-services

Kirilov, K. V. (2020, June 6). *The importance of Open source security*. CloudTweaks. Retrieved March 13, 2023, from https://cloudtweaks.com/2018/03/importance-open-source-security-vulnerabilities/

Ibrahim, J. (2021, May 17). *The evolution of web security*. Appverticals. Retrieved March 13, 2023, from https://www.appverticals.com/blog/the-evolution-of-web-security/

Input validation cheat sheet¶. Input Validation - OWASP Cheat Sheet Series. (n.d.). Retrieved March 13,  
 2023, from <https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html>

*API security: The Complete Guide to Threats, methods & tools*. Bright Security. (2022, April 21). Retrieved March 13, 2023, from https://brightsec.com/blog/api-security/#rest-api-vs-soap-security