GLOBALRAIN

Artemis Financial Vulnerability Assessment Report

Table of Contents

Document Revision History

Version	Date	Author	Comments
1.0	September 17, 2023	Matthew Pool	

Client

Developer
Matthew Pool

1. Interpreting Client Needs

Artemis Financial is a financial planning/consulting company that operates on a global level and is responsible for quite an impressive and diverse portfolio, including savings and retirement accounts, investments, and insurance. At Global Rain, our mission statement says, “Security is everyone’s responsibility”, and we truly believe that! New security vulnerabilities are continually being discovered and new, innovative and creative ways to counter them are constantly being created. Because Artemis Financial works at an international level involving financial transactions and government agencies (and other businesses), their “rest-service” software requires more than the typical RESTful web Application Programming Interface (API), as will be explained in this document, starting with domestic and international laws (including financial sector regulations), followed by general web app security vulnerability concerns, as follows:

Financial & International Government Standards & Regulatory Compliance:

- (GLBA) Gramm-Leach-Bliley Act: U.S. Law for financial info management & data protection
- (SOX) Sarbanes-Oxley Act: U.S. law for data integrity & confidentiality
- (MiFID II) Markets in Financial Instruments Directive II: European transparency regulations
- (PSD2) Revised Payment Service Directive: European directive for payment services
- (GDPR) General Data Protection Regulation: European Union law for data protection
- (FISMA) Federal Information Security Management Act: U.S. law for federal data systems
- + Other industry- and regional-specific laws (HIPAA, etc.)

VAPFD Web App Vulnerability Areas:

- Input Validation: crucial in preventing injection attacks
- APIs: provide abstraction and encapsulation for accessing methods indirectly
- Cryptography: provides data encryption and secure communication
- Client/Server: web applications use client/server architecture with both secured
- Code Error: errors in code can provide the opportunity to take advantage of exploits
- Code Quality: best practices ensure the least possibility for attacks like injection or DoS
- Encapsulation: hides implementation details from the user and reduces possible attacks

Specific Threats/Attacks:

- DoS (Denial of Service): threats that cause a software program to use an excessive number of resources, resulting in unavailable service or poor performance for legitimate users o Example: Initiating multiple connections rapidly via a script like
  - for i in {1..10000}; do curl http://target.com; done
- SQL Injection: mixing query data fragments with untrusted data to run untrusted code or access/manipulate database data o Example: inserting ' OR '1'='1 into SQL queries
- HTML Injection: web page defacement with malicious script execution o Example: inserting <script>alert('hacked')</script> into a web page
- XML Injection: unauthorized data access (or DoS) by inserting malicious script in XML parser o Example: inserting <!ENTITY xxe SYSTEM "file:///etc/passwd"> into XML data

- JSON Injection: data tamper & logic bypass by inserting script into JSON parser/web service o Example: Inserting {"isAdmin": true} into JSON payloads
- Command Injection: injecting harmful system commands
- Cross-Site Scripting (XSS): injection of malicious scripts into web pages viewed by other users, allowing access to cookies or session tokens or modifying web pages o Example: Inserting <script>stealCookies()</script> in a comment or forum post
- Cross-Site Request Forgery (CSRF): attacker tricks victim into performing actions without their consent like clicking a link they didn't want to click, resulting in unauthorized actions performed on behalf of an authenticated user o Example: Inserting an <img src="http://bank.com/withdraw?amount=1000"> tag in a forum, causing users to unknowingly make withdrawals
- Regular Expression Denial-of-Service (ReDoS): uses regular expression computation that grow exponentially with input size o Example: Inserting a complex regex like ^(a+)+$ matched with long strings of 'a's
- URL Open Redirect: misleading URLs leading to malicious sites or granting remote access o Example: Modifying http://example.com/redirect?url=trusted.com to http://example.com/redirect?url=malicious.com
- Unauthorized Access: unauthorized function/data access or misuse o Example: Exploiting a missing authentication check to access restricted API endpoints, like GET http://target.com/api/admin/users
- Misconfiguration: default or risky configurations create unnecessary exploits o Example: Leaving unsafe default configurations in a third-party plugin

Future External Threats:

As mentioned earlier, new threats are constantly being uncovered, and new types of security vulnerabilities are on the horizon. Quantum computing has been gaining traction over the years with increasing numbers of qubits (the basic unit of quantum information that uses the quantum property of superposition), allowing a quantum computer to crack encryption algorithms in a fraction of the time of a classical computer. Various complex forms of artificial intelligence are gaining popularity and already pose an existential threat to democracy and other major aspects of our lives. Deepfake technology, a subset of Generative Adversarial Networks (GANs) that creates fake media, is one way A.I. is being used by bad actors. However, governments and intelligence agencies (and other clever and cautious minds around the world) are aware of these threats and working on creative ways to combat them. Ongoing software security analyses and maintenance (updates) will be required with any lasting system, including the RESTful web application for Artemis Financial.

Modernization Requirements:

There are a few other technical requirements to consider concerning modernization of the software app. Currently, a relational SQL database is used to take advantage of ACID (Atomicity, Consistency, Isolation, Durability) transactions, which offer strong data integrity. To easily scale out, a hybrid database approach could be taken advantage of by employing a document oriented NoSQL database too, to take advantage of logging, caching, and real-time analytics. A cloud infrastructure should also be implemented to allow efficient and cost-effective performance, scalability, reliability, recoverability, and security! It is also highly recommended that accessibility considerations and integrations be made to be ADA- (Americans with Disabilities)

and WCAG- (Web Content Accessibility Guidelines) compliant. Open-source libraries, such as dependencies utilized by Maven, will be used when feasible and kept up to date. An agile Scrum framework will be utilized in continually providing security analysis and implementation that meet all security software requirements, as they continue to evolve.

2. Areas of Security

Because the Artemis Financial software app is a RESTful web API, the following VAPFD areas are of concern in terms of software security:

Controllers: handle HTTP requests and routing

- vulnerabilities: insecure endpoints, improper authentication/authorization

Data Access: uses java.sql (SQL) for accessing data in the database

- vulnerabilities: SQL/NoSQL injection, insecure data storage, lacking input validation

APIs: RESTful web API with CRUD methods to get (read) data and set (create/update) data

- vulnerabilities: improper authentication, no rate limiting, no data validation

The software application software was created using the Spring Framework and also includes the following VAPFD areas of concern:

Models: data structures and data validation throughout app

- vulnerabilities: insecure data handling/validation, weak encryption/hashing

Plugins: OWASP Dependency-Check is implemented into the software application as a plugin

Not relevant to Artemis Financial software application:

Services: REST Service Application bootstrap class created by Spring Framework to handle business logic/operations can be ignored during manual review (unless modified)

- vulnerabilities: business logic bypass, improper access control, error handling, insecure communication (unencrypted)

Views: CLI (command-line interface) output is not considered a “View” in the context of MVC architecture, and this application only uses a CLI API without additional graphical representation

- vulnerabilities: XSS, data leakage through error messages, CSRF

3. Manual Review

SECURITY VULNERABILITIES

DocData.java

- username and password are both hardcoded (and in comment)
- weak (or no) password rules
- username and password are the same
- read_document() doesn't use placeholders ('?') for key and value variables (no parameterized queries to automatically escape characters)
- auto-generated error catch block may expose sensitive information
- unused passed values (key and value)
- no SSL/TLS used for database connection (main-in-the-middle attacks)
- make id variable 'final' since an id will not change once assigned for an instance

myDateTime.java

- no visibility identifiers used for mySecond, myMinute, and myHour, resulting in packageprivate attributes that may be accessed from other classes in the package without intended authorization

customer.java

- no visibility identifier used for account_balance, resulting in package-private attributes that may be accessed from other classes in the package without intended authorization
- deposit() lacks parameter validation, which could lead to harmful inputs
- showInfo() lacks authentication/authorization and could lead to data leakage of sensitive information (account_number)

CRUDController.java

- business_name parameter not validated and could be used for SQL Injection attacks if used in database queries
- /read endpoint lacks authentication/authorization and could expose information
- no rate limiting which could lead to DoS attack
- lacking HTTPS implementation can lead to data exposure as data is transmitted in plain text
- stack traces or sensitive information may be leaked unless proper error-handling is implemented

CRUD.java

- content and content2 may be exposed unless authorization/authentication to access them are implemented and validation/sanitization are implemented

Greeting.java

- id & content may be exposed unless authorization/authentication to access them are implemented and validation/sanitization are implemented

GreetingController.java

- lack of rate limiting on number of API calls, leading to potential DoS attacks lack of authentication/authorization for the endpoint /greeting

4. Static Testing

Dependency Updates:

- org.bouncycastle:bcprov-jdk15on 1.46 -> 1.70

Plugin Updates:

- org.springframework.boot:spring-boot-maven-plugin 2.2.4.RELEASE -> 3.1.3

Plugin Validation Issues:

- org.codehaus.mojo:versions-maven-plugin:2.7
- org.apache.maven.plugins:maven-site-plugin:3.8.2

Minimum Maven 3.1.0:

- org.owasp:dependency-check-maven 5.3.0 -> 8.4.0

[WARNING] Plugin validation issues were detected in 8 plugin(s)
[WARNING]
[WARNING] * org.apache.maven.plugins:maven-resources-plugin:3.1.0
[WARNING] * org.apache.maven.plugins:maven-jar-plugin:3.1.2
[WARNING] * org.apache.maven.plugins:maven-compiler-plugin:3.8.1
[WARNING] * org.apache.maven.plugins:maven-install-plugin:2.5.2
[WARNING] * org.apache.maven.plugins:maven-clean-plugin:3.1.0
[WARNING] * org.owasp:dependency-check-maven:5.3.0
[WARNING] * org.springframework.boot:spring-boot-maven-plugin:2.2.4.RELEASE
[WARNING] * org.apache.maven.plugins:maven-surefire-plugin:2.22.2

FALSE POSITIVES (NO VULNERABILITIES)

Dependency	Vulnerability IDs↑	Package
classmate-1.5.1.jar		pkg:maven/com.fasterxml/classmate@1.5.1
jackson-annotations-2.10.2.jar		pkg:maven/com.fasterxml.jackson.core/jackson-annotations@2.10.2
jackson-core-2.10.2.jar		pkg:maven/com.fasterxml.jackson.core/jackson-core@2.10.2
jakarta.annotation-api-1.3.5.jar		pkg:maven/jakarta.annotation/jakarta.annotation-api@1.3.5
jakarta.validation-api-2.0.2.jar		pkg:maven/jakarta.validation/jakarta.validation-api@2.0.2
jboss-logging-3.4.1.Final.jar		pkg:maven/org.jboss.logging/jboss-logging@3.4.1.Final
jul-to-slf4j-1.7.30.jar		pkg:maven/org.slf4j/jul-to-slf4j@1.7.30
slf4j-api-1.7.30.jar		pkg:maven/org.slf4j/slf4j-api@1.7.30
tomcat-embed-el-9.0.30.jar		pkg:maven/org.apache.tomcat.embed/tomcat-embed-el@9.0.30

THREATS

bcprov-jdk15on-1.46.jar

Description:

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.

License:

Bouncy Castle Licence: http://www.bouncycastle.org/licence.html

File Path: C:\Users\mathy\.m2\repository\org\bouncycastle\bcprov-jdk15on\1.46\bcprov-jdk15on-1.46.jar
MD5: b94e6fef30e871f1b4117232cdc75369
SHA1: 991c96a4e31e6c19e2b9136c8955bd423f2dc4c7
SHA256:a1952237d941ef0b6122ba27b0b58de602de91c714ba3ddd4eef30ba3f5a0a67
Referenced In Project/Scope:rest-service:compile

Evidence

Identifiers

- pkg:maven/org.bouncycastle/bcprov-jdk15on@1.46 (*Confidence*:High)
- cpe:2.3:a:bouncycastle:legion-of-the-bouncy-castle-java-crytography-api:1.46:*:*:*:*:*:*:* (*Confidence*:Highest) suppress

Published Vulnerabilities

CVE-2013-1624 suppress

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.

CWE-310 Cryptographic Issues

CVSSv2:

- Base Score: MEDIUM (4.0)
- Vector: /AV:N/AC:H/Au:N/C:P/I:P/A:N

spring-boot-2.2.4.RELEASE.jar

Description:

Spring Boot

License:

Apache License, Version 2.0: https://www.apache.org/licenses/LICENSE-2.0

File Path: C:\Users\mathy\.m2\repository\org\springframework\boot\spring-boot\2.2.4.RELEASE\spring-boot-2.2.4.RELEASE.jar
MD5: 24de0cfd8ea74b903b562b43cbc5eb13
SHA1: 225a4fd31156c254e3bb92adb42ee8c6de812714
SHA256:176befc7b90e8498f44e21994a70d69ba360ef1e858ff3cea8282e802372daf2
Referenced In Project/Scope:rest-service:compile

Evidence ⊞

Related Dependencies ⊞

Identifiers

- pkg:maven/org.springframework.boot/spring-boot@2.2.4.RELEASE (*Confidence*:High)
- cpe:2.3:a:vmware:spring_boot:2.2.4:release:*:*:*:*:*:* (*Confidence*:Highest) suppress

Published Vulnerabilities ⊞

CVE-2022-27772 suppress

** UNSUPPORTED WHEN ASSIGNED ** 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. NOTE: This vulnerability only affects products and/or versions that are no longer supported by the maintainer.

CWE-668 Exposure of Resource to Wrong Sphere

CVSSv2:

- Base Score: MEDIUM (4.6)
- Vector: /AV:L/AC:L/Au:N/C:P/I:P/A:P

CVSSv3:

- Base Score: HIGH (7.8)
- Vector: /AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H

logback-core-1.2.3.jar

Description:

logback-core module

License:

http://www.eclipse.org/legal/epl-v10.html, http://www.gnu.org/licenses/old-licenses/lgpl-2.1.html

File Path: C:\Users\mathy\.m2\repository\ch\qos\logback\logback-core\1.2.3\logback-core-1.2.3.jar
MD5: 841fc80c6edff60d947a3872a2db4d45
SHA1: 864344400c3d4d92dfeb0a305dc87d953677c03c
SHA256:5946d837fe6f960c02a53eda7a6926ecc3c758bbdd69aa453ee429f858217f22
Referenced In Project/Scope:rest-service:compile

Evidence

Related Dependencies

Identifiers

- pkg:maven/ch.qos.logback/logback-core@1.2.3 (*Confidence*:High)
- cpe:2.3:a:qos:logback:1.2.3:*:*:*:*:*:*:* (*Confidence*:Highest) suppress

Published Vulnerabilities

CVE-2021-42550 suppress

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.

CWE-502 Deserialization of Untrusted Data

CVSSv2:

- Base Score: HIGH (8.5)
- Vector: /AV:N/AC:M/Au:S/C:C/I:C/A:C

CVSSv3:

- Base Score: MEDIUM (6.6)
- Vector: /AV:N/AC:H/PR:H/UI:N/S:U/C:H/I:H/A:H

log4j-api-2.12.1.jar

Description:

The Apache Log4j API

License:

https://www.apache.org/licenses/LICENSE-2.0.txt

File Path: C:\Users\mathy\.m2\repository\org\apache\logging\log4j\log4j-api\2.12.1\log4j-api-2.12.1.jar
MD5: 4a6f276d4fb426c8d489343c0325bb75
SHA1: a55e6d987f50a515c9260b0451b4fa217dc539cb
SHA256:429534d03bdb728879ab551d469e26f6f7ff4c8a8627f59ac68ab6ef26063515
Referenced In Project/Scope:rest-service:compile

Evidence

Related Dependencies

Identifiers

- pkg:maven/org.apache.logging.log4j/log4j-api@2.12.1 (*Confidence*:High)
- cpe:2.3:a:apache:log4j:2.12.1:*:*:*:*:*:*:* (*Confidence*:Highest) suppress

Published Vulnerabilities

CVE-2020-9488 suppress

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. Fixed in Apache Log4j 2.12.3 and 2.13.1

CWE-295 Improper Certificate Validation

CVSSv2:

- Base Score: MEDIUM (4.3)
- Vector: /AV:N/AC:M/Au:N/C:P/I:P/A:N

CVSSv3:

- Base Score: LOW (3.7)
- Vector: /AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N

snakeyaml-1.25.jar

Description:

YAML 1.1 parser and emitter for Java

License:

Apache License, Version 2.0: http://www.apache.org/licenses/LICENSE-2.0.txt

File Path: C:\Users\mathy\.m2\repository\org\yaml\snakeyaml\1.25\snakeyaml-1.25.jar
MD5: 6f7d5b8f596047aae07a3bf6f23a0bf2
SHA1: 8b6e01ef661d8378ae6dd7b511a7f2a33fae1421
SHA256:b50ef33187e7dc922b26dbe4dd0fdb3a9cf349e75a08b95269901548eee546eb
Referenced In Project/Scope:rest-service:runtime

Evidence

Identifiers

- pkg:maven/org.yaml/snakeyaml@1.25 (*Confidence*:High)
- cpe:2.3:a:snakeyaml_project:snakeyaml:1.25:*:*:*:*:*:*:* (*Confidence*:Highest) suppress
- cpe:2.3:a:yaml_project:yaml:1.25:*:*:*:*:*:*:* (*Confidence*:Highest) suppress

Published Vulnerabilities

CVE-2017-18640 suppress

The Alias feature in SnakeYAML before 1.26 allows entity expansion during a load operation, a related issue to CVE-2003-1564.

CWE-776 Improper Restriction of Recursive Entity References in DTDs ('XML Entity Expansion')

CVSSv2:
- Base Score: MEDIUM (5.0)
- Vector: /AV:N/AC:L/Au:N/C:N/I:N/A:P

CVSSv3:
- Base Score: HIGH (7.5)
- Vector: /AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

jackson-databind-2.10.2.jar

Description:

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

License:

http://www.apache.org/licenses/LICENSE-2.0.txt

File Path: C:\Users\mathy\.m2\repository\com\fasterxml\jackson\core\jackson-databind\2.10.2\jackson-databind-2.10.2.jar
MD5: 057751b4e2dd1104be8caad6e9a3e589
SHA1: 0528de95f198afafbcfb0c09d2e43b6e0ea663ec
SHA256:42c25644e35fadfbded1b7f35a8d1e70a86737f190e43aa2c56cea4b96cbda88
Referenced In Project/Scope:rest-service:compile

Evidence

Identifiers

- pkg:maven/com.fasterxml.jackson.core/jackson-databind@2.10.2 (*Confidence*:High)
- cpe:2.3:a:fasterxml:jackson-databind:2.10.2:*:*:*:*:*:*:* (*Confidence*:Highest) suppress

Published Vulnerabilities

CVE-2020-25649 suppress

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.

CWE-611 Improper Restriction of XML External Entity Reference ('XXE')

CVSSv2:

- Base Score: MEDIUM (5.0)
- Vector: /AV:N/AC:L/Au:N/C:N/I:N/A:N

CVSSv3:

- Base Score: HIGH (7.5)
- Vector: /AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:H/A:N

tomcat-embed-core-9.0.30.jar

Description:

Core Tomcat implementation

License:

Apache License, Version 2.0: http://www.apache.org/licenses/LICENSE-2.0.txt

File Path: C:\Users\mathy\.m2\repository\org\apache\tomcat\embed\tomcat-embed-core\9.0.30\tomcat-embed-core-9.0.30.jar
MD5: f9e49f66756f133157f19e617af26ffe
SHA1: ad32909314fe2ba02cec036434c0addd19bcc580
SHA256:b1415eecbc9f14e3745c1bfd41512a1b8e1af1332a7beaed4be30b2e0ba7b330
Referenced In Project/Scope:rest-service:compile

Evidence

Related Dependencies

Identifiers

- pkg:maven/org.apache.tomcat.embed/tomcat-embed-core@9.0.30 (*Confidence*:High)
- cpe:2.3:a:apache:tomcat:9.0.30:*:*:*:*:*:*:* (*Confidence*:Highest) suppress
- cpe:2.3:a:apache_tomcat:apache_tomcat:9.0.30:*:*:*:*:*:*:* (*Confidence*:Highest) suppress

Published Vulnerabilities

CVE-2019-17569 suppress

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. Such a reverse proxy is considered unlikely.

CWE-444 Inconsistent Interpretation of HTTP Requests ('HTTP Request Smuggling')

CVSSv2:

- Base Score: MEDIUM (5.8)
- Vector: /AV:N/AC:M/Au:N/C:P/I:P/A:N

CVSSv3:

- Base Score: MEDIUM (4.8)
- Vector: /AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:L/A:N

hibernate-validator-6.0.18.Final.jar

Description:

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

License:

http://www.apache.org/licenses/LICENSE-2.0.txt

File Path: C:\Users\mathy\.m2\repository\org\hibernate\validator\hibernate-validator\6.0.18.Final\hibernate-validator-6.0.18.Final.jar
MD5: d3eeb4f1bf013d939b86dfc34b0c6a5d
SHA1: 7fd00bcd87e14b6ba66279282ef15efa30dd2492
SHA256:79fb11445bc48e1ea6fb259e825d58b3c9a5fa2b7e3c9527e41e4aeda82de907
Referenced In Project/Scope:rest-service:compile

Evidence

Identifiers

- pkg:maven/org.hibernate.validator/hibernate-validator@6.0.18.Final (*Confidence*:High)
- cpe:2.3:a:redhat:hibernate_validator:6.0.18:*:*:*:*:*:*:* (*Confidence*:Highest) suppress

Published Vulnerabilities

CVE-2020-10693 suppress

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.

CWE-20 Improper Input Validation

CVSSv2:

- Base Score: MEDIUM (5.0)
- Vector: /AV:N/AC:L/Au:N/C:N/I:N/A:N

CVSSv3:

- Base Score: MEDIUM (5.3)
- Vector: /AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:L/A:N

spring-web-5.2.3.RELEASE.jar

Description:

Spring Web

License:

Apache License, Version 2.0: https://www.apache.org/licenses/LICENSE-2.0

File Path: C:\Users\mathy\.m2\repository\org\springframework\spring-web\5.2.3.RELEASE\spring-web-5.2.3.RELEASE.jar
MD5: a89d66690cd14159aa6ac1e875e54411
SHA1: dd386a02e40b915ab400a3bf9f586d2dc4c0852c
SHA256:25d264969c624cb8103a7f2b36b148ea1be8b87780c4758e7f9a6e2bc8416d76
Referenced In Project/Scope:rest-service:compile

Evidence

Identifiers

- pkg:maven/org.springframework/spring-web@5.2.3.RELEASE (*Confidence*:High)
- cpe:2.3:a:pivotal_software:spring_framework:5.2.3:release:*:*:*:*:*:* (*Confidence*:Highest) suppress
- cpe:2.3:a:springsource:spring_framework:5.2.3:release:*:*:*:*:*:* (*Confidence*:Highest) suppress

Published Vulnerabilities

CVE-2016-1000027 (OSSINDEX) suppress

Pivotal Spring Framework through 5.3.16 suffers from a potential remote code execution (RCE) issue if used for Java deserialization of untrusted data. Depending on how the library is implemented within a product, this issue may or not occur, and authentication may be required. NOTE: the vendor's position is that untrusted data is not an intended use case. The product's behavior will not be changed because some users rely on deserialization of trusted data.

CWE-502 Deserialization of Untrusted Data

CVSSv2:

- Base Score: HIGH (9.8)
- Vector: /AV:N/AC:L/Au:/C:H/I:H/A:H

spring-beans-5.2.3.RELEASE.jar

Description:

Spring Beans

License:

Apache License, Version 2.0: https://www.apache.org/licenses/LICENSE-2.0

File Path: C:\Users\mathy\.m2\repository\org\springframework\spring-beans\5.2.3.RELEASE\spring-beans-5.2.3.RELEASE.jar
MD5: b64e8da412c3b6100f4bc0f54325d44f
SHA1: 0250c8c641433dc06b1b44e4563fa08a2fbf8954
SHA256:d3083070ad4eaf32e003b86ca0e7c0cb4fd2819800fef86ceb1043d387c14e2d
Referenced In Project/Scope:rest-service:compile

Evidence

Identifiers

- pkg:maven/org.springframework/spring-beans@5.2.3.RELEASE (*Confidence*:High)
- cpe:2.3:a:pivotal_software:spring_framework:5.2.3:release:*:*:*:*:*:* (*Confidence*:Highest) suppress
- cpe:2.3:a:springsource:spring_framework:5.2.3:release:*:*:*:*:*:* (*Confidence*:Highest) suppress

Published Vulnerabilities

CVE-2022-22965 (OSSINDEX) suppress

A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. The specific exploit requires the application to run on Tomcat as a WAR deployment. If the application is deployed as a Spring Boot executable jar, i.e. the default, it is not vulnerable to the exploit. However, the nature of the vulnerability is more general, and there may be other ways to exploit it.

CWE-94 Improper Control of Generation of Code ('Code Injection')

CVSSv2:

- Base Score: HIGH (9.8)
- Vector: /AV:N/AC:L/Au:/C:H/I:H/A:H

spring-webmvc-5.2.3.RELEASE.jar

Description:

Spring Web MVC

License:

Apache License, Version 2.0: https://www.apache.org/licenses/LICENSE-2.0

File Path: C:\Users\mathy\.m2\repository\org\springframework\spring-webmvc\5.2.3.RELEASE\spring-webmvc-5.2.3.RELEASE.jar
MD5: 867cc7369d453637b5042ee4d6931a78
SHA1: 745a62502023d2496b565b7fe102bb1ee229d6b7
SHA256:b3b0a2477e67b050dd5c08dc96e76db5950cbccba075e782c24f73eda49a0160
Referenced In Project/Scope:rest-service:compile

Evidence

Identifiers

- pkg:maven/org.springframework/spring-webmvc@5.2.3.RELEASE (*Confidence*:High)
- cpe:2.3:a:pivotal_software:spring_framework:5.2.3:release:*:*:*:*:*:* (*Confidence*:Highest) suppress
- cpe:2.3:a:springsource:spring_framework:5.2.3:release:*:*:*:*:*:* (*Confidence*:Highest) suppress

Published Vulnerabilities

CVE-2021-22060 (OSSINDEX) suppress

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. This is a follow-up to CVE-2021-22096 that protects against additional types of input and in more places of the Spring Framework codebase.

CWE-117 Improper Output Neutralization for Logs

CVSSv2:

- Base Score: MEDIUM (4.3)
- Vector: /AV:N/AC:L/Au:/C:N/I:L/A:N

spring-context-5.2.3.RELEASE.jar

Description:

Spring Context

License:

Apache License, Version 2.0: https://www.apache.org/licenses/LICENSE-2.0

File Path: C:\Users\mathy\.m2\repository\org\springframework\spring-context\5.2.3.RELEASE\spring-context-5.2.3.RELEASE.jar
MD5: a5ba542a35f3c9fca630df1715ccf325
SHA1: 7750c95c96c7a1885c8b1b503ba915bc33ca579a
SHA256:82c625cffed80685b153700359a6c6d5c91018069a0171cf21a7defb0267e993
Referenced In Project/Scope:rest-service:compile

Evidence

Identifiers

- pkg:maven/org.springframework/spring-context@5.2.3.RELEASE (*Confidence*:High)
- cpe:2.3:a:pivotal_software:spring_framework:5.2.3:release:*:*:*:*:*:* (*Confidence*:Highest) suppress
- cpe:2.3:a:springsource:spring_framework:5.2.3:release:*:*:*:*:*:* (*Confidence*:Highest) suppress

Published Vulnerabilities

CVE-2022-22968 (OSSINDEX) suppress

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.

CWE-178 Improper Handling of Case Sensitivity

CVSSv2:

- Base Score: MEDIUM (5.3)
- Vector: /AV:N/AC:L/Au:/C:N/I:L/A:N

spring-expression-5.2.3.RELEASE.jar

Description:

Spring Expression Language (SpEL)

License:

Apache License, Version 2.0: https://www.apache.org/licenses/LICENSE-2.0

File Path: C:\Users\mathy\.m2\repository\org\springframework\spring-expression\5.2.3.RELEASE\spring-expression-5.2.3.RELEASE.jar
MD5: f2d2fe0e4f9b9b23b03d07839393de5a
SHA1: d0c6bb10758805b2153c589686b8045554bfac2d
SHA256:1ba798e1f4da9e5ad68e67d7e7abe39f141674762c8755d952edeb0380d384b9
Referenced In Project/Scope:rest-service:compile

Evidence

Identifiers

- pkg:maven/org.springframework/spring-expression@5.2.3.RELEASE (*Confidence*:High)
- cpe:2.3:a:pivotal_software:spring_framework:5.2.3:release:*:*:*:*:*:* (*Confidence*:Highest) suppress
- cpe:2.3:a:springsource:spring_framework:5.2.3:release:*:*:*:*:*:* (*Confidence*:Highest) suppress

Published Vulnerabilities

CVE-2022-22950 (OSSINDEX) suppress

n 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.

CWE-770 Allocation of Resources Without Limits or Throttling

CVSSv2:

- Base Score: MEDIUM (6.5)
- Vector: /AV:N/AC:L/Au:/C:N/I:N/A:H

5. Mitigation Plan

As previously stated, the Artemis Financial software application must adhere to strict guidelines set forth by various governmental and industry regulations. On top of this, RESTful web applications are prone to certain security vulnerabilities and must be kept up to date, and security requirements must continually be checked, as new vulnerabilities and threats are discovered.

The following changes should be made to the rest-service package classes listed below:

DocData.java

- remove hardcoded username and password and comment
- pass in the username and access salted SHA-256 encrypted password using AES and POST, along with masking, as well as forcing the password to be different than the username
- input validation annotations and custom logic to enforce password rules to require at least 8 digits and 1 digit and special character
- use parameterized queries with placeholders ('?') for key and value variables in read_document() and implement their use (or completely remove if unused)
- use global and localized error catching blocks that log and return purged information
- use SSL (TLS) for all database connections

myDateTime.java

- make mySecond, myMinute, and myHour variables 'private'

customer.java

- make account_balance 'private'
- make account_number 'final'
- add Spring Security dependency for authentication/authorization check for showInfo()
- validate parameters passed to deposit()

CRUDController.java

- add validation for business_name parameter
- add Spring Security dependency for authentication/authorization for endpoint
- use Bucket4j library for rate limiting
- use global and localized error catching blocks that log and return purged information

CRUD.java

- add Spring Security dependency for authorization/authentication to access content and content2
- implement validation/sanitization for content and content2

Greeting.java

- add Spring Security dependency for authorization/authentication to access id and content
- implement validation/sanitization for id and content

GreetingController.java

- use Bucket4j library for rate limiting

- add Spring Security dependency for authentication/authorization for /greeting endpoint
  Dependency Updates:
- org.bouncycastle:bcprov-jdk15on 1.46 -> 1.70

Plugin Updates:
- org.springframework.boot:spring-boot-maven-plugin 2.2.4 RELEASE -> 3.1.3

Plugin Validation Issues:
- org.codehaus.mojo:versions-maven-plugin:2.7
- org.apache.maven.plugins:maven-site-plugin:3.8.2

Other Plugin Issues:
- org.apache.maven.plugins:maven-resources-plugin:3.1.0
- org.apache.maven.plugins:maven-jar-plugin:3.1.2
- org.apache.maven.plugins:maven-compiler-plugin:3.8.1
- org.apache.maven.plugins:maven-install-plugin:2.5.2
- org.apache.maven.plugins:maven-clean-plugin:3.1.0
- org.owasp:dependency-check-maven:5.3.0
- org.springframework.boot:spring-boot-maven-plugin:2.2.4.RELEASE
  org.apache.maven.plugins:maven-surefire-plugin:2.22.2

Minimum Maven 3.1.0:
- org.owasp:dependency-check-maven 5.3.0 -> 8.4.0

Based on the previous results, the following steps should be taken to address vulnerabilities:

GLOBAL CODE MODIFICATIONS

- implement HTTPS into application configuration settings
- implement Spring global error catching

UPDATES

- Update Spring Framework
- Update OWASP Dependency-Check plugin
- Update out-of-date dependencies
- Update out-of-date libraries and other plugins
- Apply any applicable vendor security patches
- Run OWASP (Maven) Dependency-Check & check results

CVE DEPENDENCY UPDATES

Apply the listed strategies for any of the following CVE Vulnerabilities that remain:

- CVE-2013-1624: Update org.bouncycastle:bcprov-jdk15on 1.46 -> 1.70
- CVE-2022-27772: Update org.springframework.boot:spring-boot-mavenplugin:2.2.4.RELEASE
- CVE-2021-42550: Update org.apache.maven.plugins:maven-site-plugin:3.8.2
- CVE-2017-18640: Update org.springframework.boot:spring-boot-mavenplugin:2.2.4.RELEASE
- CVE-2020-25649: Update org.springframework.boot:spring-boot-mavenplugin:2.2.4.RELEASE
- CVE-2019-17569: Update org.springframework.boot:spring-boot-mavenplugin:2.2.4.RELEASE
- CVE-2020-10693: Update to patched version 6.15.Final (or above)
- CVE-2016-1000027 (OSSINDEX): Update spring-web (org.springframework) dependency in pom.xml
- CVE-2022-22965 (OSSINDEX): Update spring-bean (org.springframework) dependency in pom.xml
- CVE-2021-22060 (OSSINDEX): Update spring-webmvc (org.springframework) dependency in pom.xml
- CVE-2022-22968 (OSSINDEX): Update spring-context (org.springframework) dependency in pom.xml
- CVE-2022-22950 (OSSINDEX): Update spring-expression (org.springframework) dependency in pom.xml

Save pom.xml, create a Maven clean installation, and verify security vulnerabilities have been fixed by the preceding steps.

Various tools from OWASP and other open-sources may be used to perform final vulnerability assessment and analysis.

Sources:

- Common Vulnerabilities and Exposures (CVE).(2023). https://cve.mitre.org/cve/search_cve_list.html
- Common Weakness Enumeration (CWE). (2023). https://cwe.mitre.org/index.html
- National Vulnerability Database (NVD). (2023).
- OWASP Secure Coding Practices Checklist: https://owasp.org/www-pdfarchive/OWASP_SCP_Quick_Reference_Guide_v2.pdf https://nvd.nist.gov/vuln/search
- Secure Coding Guidelines for Java SE: https://www.oracle.com/java/technologies/javase/seccodeguide.html