****

# 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** | **3/21/2021** | **Victor Feight** | **Initial delivery** |

## 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

Victor Feight

## 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?

Points of consideration: This company deals directly with customers delivering highly sensitive data including financial plans for savings, retirement, investments, and insurance. Any of these plans may contain data such as social security, financial information, addresses, numbers that shouldn’t be leaked or able to be extracted. In accordance with OWASP, encryption is highly recommended for transmission of such sensitive information, by ensuring a secure protocol such as HTTPS is used to secure data communication between server and client; this prevents man in the middle attacks. Any server side code should be protected, no information should be stored or accessible from logs. Secure communication is a necessity to this company.

* Are there any international transactions that the company produces?

This company Artemis Financial specializes in financial software using a RESTful API; thus, international transactions should be taken into consideration. This might entail ensuring data validation is conducted on the server, encoding data to a common character set before validating, and determining UTF-8 support for international characters.

* Are there governmental restrictions about secure communications to consider?

Indeed, when dealing with such highly sensitive financial documents and data, there are government restrictions to consider. Financial privacy laws in the US are enacted at the federal level, though regulation varies from state to state. One such example is the “Right to Financial Privacy Act” from 1978, requiring legal notice for a gov’t to view a person’s financial information; such policies must be taken into consideration for this software, and financial laws may differ by country as well. Another example is the Gramm-Leach-Bliley Act which requires financial institutions to explain their information sharing practices to customers, and to safeguard their sensitive data.

* What external threats might be present now and in the immediate future?

OWASP threats listed include categories of Data Validation, Authentication and Password concerns (don’t store sensitive data in POST for example), Access/Session management, Sensitive information storage, System configuration management , Database security, and File/Memory management. Some important threats to protect against always include: web application attacks such as SQL/XML injection, DDoS, or unvalidated redirects. DDoS attacks (denial of service) can be very costly for company with highly sensitive data, disrupting business flow. Backdoors are threats used to gain remote access to the network, which should be monitored. External vendors may have security threats within their network that’s not controllable within ours, though they may affect us. Lastly, since this is a global company, there are global operational risk increases; threats to a financial application is highly targeted in a globalized setting, and security measures should be accounted for across all regions and vendors.

* What are the “modernization” requirements that must be considered, such as the role of open source libraries and evolving web application technologies?

In combatting cross-site scripting and other threats, Iron-Clad java recommends using time-tested libraries to save developers time, providing good defense; this is preferable to rolling your own defense solution. Some examples include: OWASP Java Encoder, JQuery Encoder, OWASP HTML Sanitizer, or jsJHtmlSanitizer. In addition, becoming familiar with Apache shiro for authentication and authorization, Logback for logging, jdeps for library dependency detection, or FindBugs for integrating into IDE or a continuous integration solution.

Another example of emerging technology include the blockchain or the Internet of Things (IoT), such emerging technologies can only be regulated by meeting cybersecurity requirements and ensuring multi-factor authentication along with other OWASP standard recommendations.

## 2. Areas of Security

Referring to the Vulnerability Assessment Process Flow Diagram, identify which areas of security are applicable to Artemis Financial’s software application. Justify your reasoning for why each area is relevant to the software application.

1. Input validation is required for a RESTful API, it’s included in OWASPS security guidelines. Specifically, “whenever possible, processing untrusted input should be avoided”, specifically for password input, databases, filestreams, hazardous characters, data length, or range.
2. Secure API interactions are a necessity, as we will be dealing with a REST service, and thus providing secure HTTP endpoints is necessary. This includes protection for sensitive authentication credentials such as API keys or JSON with user data. Access Control is also important for securing API with user authentication, authorization, and session management.
3. As we are dealing with the transmission of Sensitive Information (financial), OWASP recommends implementing encryption. Specifically, server side code should be protected and highly sensitive information encrypted and on server-side.
4. This application shall be a Web-based software with a RESTful interface. As such, this entails a client-server model with a database. Store sensitive information server-side only, and keep none in logs; protect server side code from being downloaded, and utilize principle of least privilege in system configuration.
5. Encapsulation: Java Security Guidelines recommends setting fields private and avoiding accessors, as this can lead to security attacks. OWASP recommends encapsulating the library and exposing only required behavior into our software; when choosing a library for our RESTful interface, encapsulation must be taken into consideration.

## 3. Manual Review

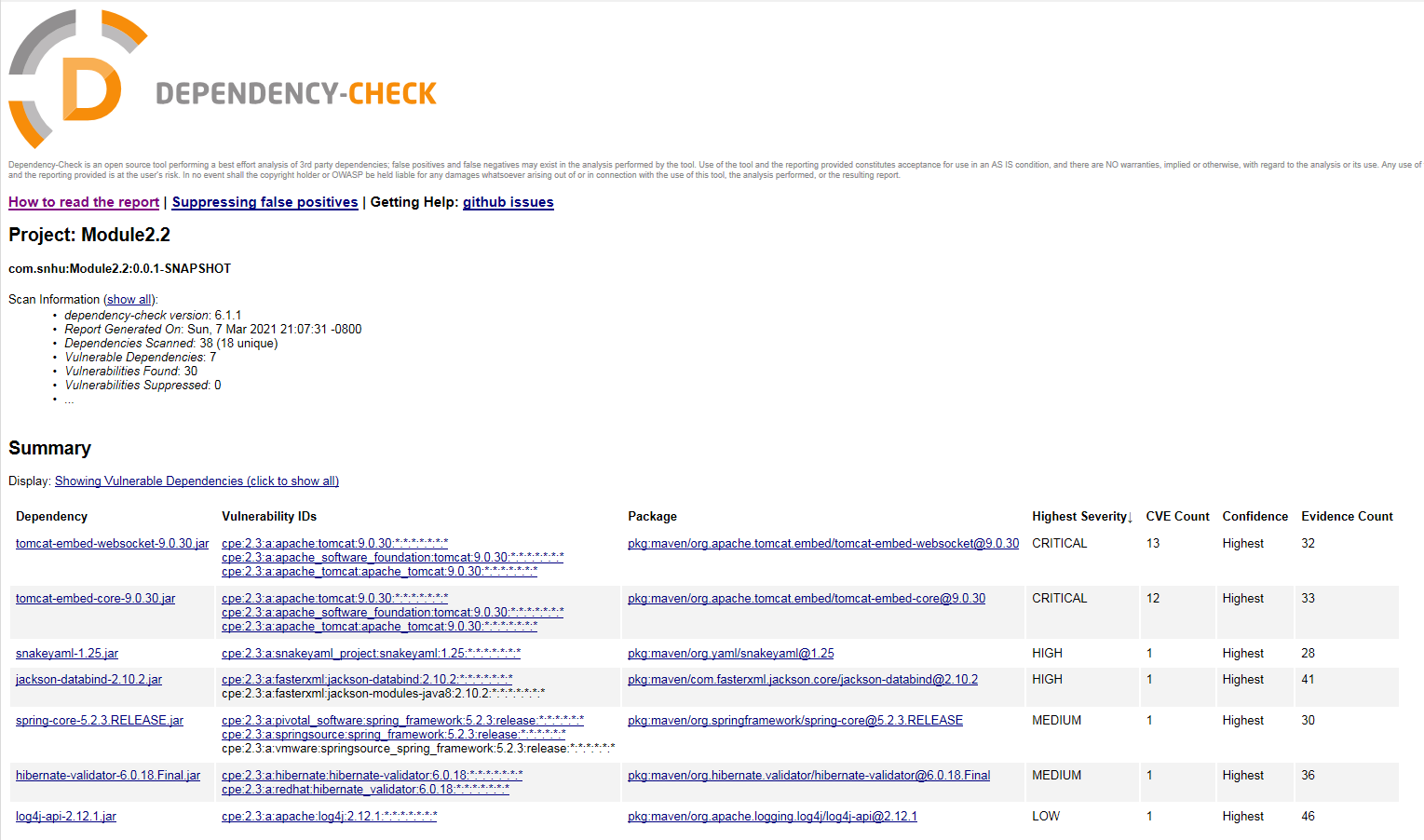
Continue working through the Vulnerability Assessment Process Flow Diagram. Identify all vulnerabilities in the code base by manually inspecting the code.

* GreetingController class: Java’s ExpressionParser object should have input validation. The use of SpelExpressionParser.parseExpression() with a dynamic parameter allows the parser to parse expressions placed between curley brackets, a potential vulnerability.
* If the class has no plans for inheritance, declare it *final* to prevent a method being maliciously overridden.
* Should include boundary analysis for myArray – an attacker could cause an out of bounds error by passing id some huge value. It also appears that there’s no checking for an Array index out of bounds error, which should be implemented.
* Consider implementing ROLES for each API endpoint, allowing users which priveleges to access those endpoints using @Secured annotation.
* Upon running the software, noted Spring version v2.2.4 was in use rather than the newest 2.4.3. Consider updating to mitigate any security vulnerabilities.

## 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



Vulnerable dependencies:

* [tomcat-embed-websocket-9.0.30.jar](file:///C:\Users\Vic\eclipse-workspace\Module2.1_Test\target\dependency-check-report.html#l18_33157f6bc5bfd03380ebb5ac476db0600a04168d)
  + Package: [pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket@9.0.30](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-websocket@9.0.30?utm_source=dependency-check&utm_medium=integration&utm_content=6.1.1)
  + CVE / Evidence Count: 13 / 32
  + Severity: Critical
  + Description: Core Tomcat implementation
  + Vulnerability IDs: [cpe:2.3:a:apache:tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Aapache&cpe_product=cpe%3A%2F%3Aapache%3Atomcat&cpe_version=cpe%3A%2F%3Aapache%3Atomcat%3A9.0.30)  
    [cpe:2.3:a:apache\_software\_foundation:tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Aapache_software_foundation&cpe_product=cpe%3A%2F%3Aapache_software_foundation%3Atomcat&cpe_version=cpe%3A%2F%3Aapache_software_foundation%3Atomcat%3A9.0.30)  
    [cpe:2.3:a:apache\_tomcat:apache\_tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Aapache_tomcat&cpe_product=cpe%3A%2F%3Aapache_tomcat%3Aapache_tomcat&cpe_version=cpe%3A%2F%3Aapache_tomcat%3Aapache_tomcat%3A9.0.30)
* [tomcat-embed-core-9.0.30.jar](file:///C:\Users\Vic\eclipse-workspace\Module2.1_Test\target\dependency-check-report.html#l16_ad32909314fe2ba02cec036434c0addd19bcc580)
  + Package: [pkg:maven/org.apache.tomcat.embed/tomcat-embed-core@9.0.30](https://ossindex.sonatype.org/component/pkg:maven/org.apache.tomcat.embed/tomcat-embed-core@9.0.30?utm_source=dependency-check&utm_medium=integration&utm_content=6.1.1)
  + CVE / Evidence Count: 12 / 33
  + Severity: Critical
  + Description: Core Tomcat implementation
  + Vulnerability IDs: [cpe:2.3:a:apache:tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Aapache&cpe_product=cpe%3A%2F%3Aapache%3Atomcat&cpe_version=cpe%3A%2F%3Aapache%3Atomcat%3A9.0.30)  
    [cpe:2.3:a:apache\_software\_foundation:tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Aapache_software_foundation&cpe_product=cpe%3A%2F%3Aapache_software_foundation%3Atomcat&cpe_version=cpe%3A%2F%3Aapache_software_foundation%3Atomcat%3A9.0.30)  
    [cpe:2.3:a:apache\_tomcat:apache\_tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Aapache_tomcat&cpe_product=cpe%3A%2F%3Aapache_tomcat%3Aapache_tomcat&cpe_version=cpe%3A%2F%3Aapache_tomcat%3Aapache_tomcat%3A9.0.30)
* [snakeyaml-1.25.jar](file:///C:\Users\Vic\eclipse-workspace\Module2.1_Test\target\dependency-check-report.html#l13_8b6e01ef661d8378ae6dd7b511a7f2a33fae1421)
  + Package: [pkg:maven/org.yaml/snakeyaml@1.25](https://ossindex.sonatype.org/component/pkg:maven/org.yaml/snakeyaml@1.25?utm_source=dependency-check&utm_medium=integration&utm_content=6.1.1)
  + CVE / Evidence Count: 1 / 28
  + Severity: High
  + Description: YAML 1.1 parser and emitter for Java
  + Vulnerability IDs:  [cpe:2.3:a:snakeyaml\_project:snakeyaml:1.25:\*:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Asnakeyaml_project&cpe_product=cpe%3A%2F%3Asnakeyaml_project%3Asnakeyaml&cpe_version=cpe%3A%2F%3Asnakeyaml_project%3Asnakeyaml%3A1.25)
* [jackson-databind-2.10.2.jar](file:///C:\Users\Vic\eclipse-workspace\Module2.1_Test\target\dependency-check-report.html#l4_0528de95f198afafbcfb0c09d2e43b6e0ea663ec)
  + Package: [pkg:maven/com.fasterxml.jackson.core/jackson-databind@2.10.2](https://ossindex.sonatype.org/component/pkg:maven/com.fasterxml.jackson.core/jackson-databind@2.10.2?utm_source=dependency-check&utm_medium=integration&utm_content=6.1.1)
  + CVE / Evidence Count: 1 / 41
  + Severity: High
  + Description: General data-binding functionality for Jackson: works on core streaming API
  + Vulnerability IDs: [cpe:2.3:a:fasterxml:jackson-databind:2.10.2:\*:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Afasterxml&cpe_product=cpe%3A%2F%3Afasterxml%3Ajackson-databind&cpe_version=cpe%3A%2F%3Afasterxml%3Ajackson-databind%3A2.10.2)  
    cpe:2.3:a:fasterxml:jackson-modules-java8:2.10.2:\*:\*:\*:\*:\*:\*:\*
* [spring-core-5.2.3.RELEASE.jar](file:///C:\Users\Vic\eclipse-workspace\Module2.1_Test\target\dependency-check-report.html#l15_3734223040040e8c3fecd5faa3ae8a1ed6da146b)
  + Package: [pkg:maven/org.springframework/spring-core@5.2.3.RELEASE](https://ossindex.sonatype.org/component/pkg:maven/org.springframework/spring-core@5.2.3.RELEASE?utm_source=dependency-check&utm_medium=integration&utm_content=6.1.1)
  + CVE / Evidence Count: 1 / 30
  + Severity: Medium
  + Description: Spring Core
  + Vulnerability IDs:  [cpe:2.3:a:pivotal\_software:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Apivotal_software&cpe_product=cpe%3A%2F%3Apivotal_software%3Aspring_framework&cpe_version=cpe%3A%2F%3Apivotal_software%3Aspring_framework%3A5.2.3)  
    [cpe:2.3:a:springsource:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Aspringsource&cpe_product=cpe%3A%2F%3Aspringsource%3Aspring_framework&cpe_version=cpe%3A%2F%3Aspringsource%3Aspring_framework%3A5.2.3)  
    cpe:2.3:a:vmware:springsource\_spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*
* [hibernate-validator-6.0.18.Final.jar](file:///C:\Users\Vic\eclipse-workspace\Module2.1_Test\target\dependency-check-report.html#l2_7fd00bcd87e14b6ba66279282ef15efa30dd2492)
  + Package:  [pkg:maven/org.hibernate.validator/hibernate-validator@6.0.18.Final](https://ossindex.sonatype.org/component/pkg:maven/org.hibernate.validator/hibernate-validator@6.0.18.Final?utm_source=dependency-check&utm_medium=integration&utm_content=6.1.1)
  + CVE / Evidence Count: 1 / 36
  + Severity: Medium
  + Description: Hibernate's Bean Validation (JSR-380) reference implementation.
  + Vulnerability IDs: [cpe:2.3:a:hibernate:hibernate-validator:6.0.18:\*:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Ahibernate&cpe_product=cpe%3A%2F%3Ahibernate%3Ahibernate-validator&cpe_version=cpe%3A%2F%3Ahibernate%3Ahibernate-validator%3A6.0.18)  
    [cpe:2.3:a:redhat:hibernate\_validator:6.0.18:\*:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Aredhat&cpe_product=cpe%3A%2F%3Aredhat%3Ahibernate_validator&cpe_version=cpe%3A%2F%3Aredhat%3Ahibernate_validator%3A6.0.18)
* [log4j-api-2.12.1.jar](file:///C:\Users\Vic\eclipse-workspace\Module2.1_Test\target\dependency-check-report.html#l9_a55e6d987f50a515c9260b0451b4fa217dc539cb)
  + Package:  [pkg:maven/org.apache.logging.log4j/log4j-api@2.12.1](https://ossindex.sonatype.org/component/pkg:maven/org.apache.logging.log4j/log4j-api@2.12.1?utm_source=dependency-check&utm_medium=integration&utm_content=6.1.1)
  + CVE / Evidence Count: 1 / 46
  + Severity: Low
  + Description: The Apache Log4j API
  + Vulnerability IDs: [cpe:2.3:a:apache:log4j:2.12.1:\*:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Aapache&cpe_product=cpe%3A%2F%3Aapache%3Alog4j&cpe_version=cpe%3A%2F%3Aapache%3Alog4j%3A2.12.1)

## 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.

Manual review: Implement input validation on ExpressionParser, boundary analysis on myArray, set class to private, update Spring version, implement roles.

Static Analysis Mitigation Plan:

[**CVE-2019-17569**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2019-17569)

Refactoring present in Apache Tomcat 9.0.28 to 9.0.30 introduced a regression, resulting in invalid Transfer-Encoding headers being incorrectly processed. This lead to HTTP Request smuggling possibility. Solution is to upgrade apache tomcat to 9.0.31+.

[**CVE-2020-13934**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-13934)

An h2c direct connection to ApacheTomcat 9.0.0.M5 to 9.0.36 did not release the HTTP/1.1 processor, after upgrading to HTTP/2. This could lead to Denial of Service and OutOfMemoryException. Solution is to upgrade apache tomcat to 9.0.37.

[tomcat-embed-core-9.0.30.jar](file:///C:\Users\Vic\eclipse-workspace\Module2.1_Test\target\dependency-check-report.html#l16_ad32909314fe2ba02cec036434c0addd19bcc580)

[**CVE-2020-11996**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-11996)

A sequence of specially crafted HTTP/2 requests sent to Apache Tomcat 9.0.0.M1 to 9.0.35 could trigger high CPU for several seconds, causing the server to become unresponsive. Solution: Apache-tomcat upgrade to 9.0.36.

[**CVE-2020-1938**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-1938)

A critical Apache Tomcat vulnerability: “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.” These connections can be exploited by an attacker.

A workaround: Opening "server.xml" in a text editor and commenting out "protocol="AJP/1.3"" connector. Restart SVF services.

If AJP protocol used: Specify IP to allow access from secure server only (loop-back 127.0.0.1)

[snakeyaml-1.25.jar](file:///C:\Users\Vic\eclipse-workspace\Module2.1_Test\target\dependency-check-report.html#l13_8b6e01ef661d8378ae6dd7b511a7f2a33fae1421)

[**CVE-2017-18640**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2017-18640)

“The Alias feature in SnakeYAML 1.18 allows entity expansion during a load operation, a related issue to CVE-2003-1564.”

Fix: Developer has fixed this in version 1.26+. Please update to that version.

[jackson-databind-2.10.2.jar](file:///C:\Users\Vic\eclipse-workspace\Module2.1_Test\target\dependency-check-report.html#l4_0528de95f198afafbcfb0c09d2e43b6e0ea663ec)

[**CVE-2020-25649**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-25649)

A flaw found in FasterXML Jackson Databind – entity expansion not secured properly, leaving vulnerability to XML entity attacks, risking data integrity.

Fix: Upgrading to 2.10.5.1 will solve the problem.

[spring-core-5.2.3.RELEASE.jar](file:///C:\Users\Vic\eclipse-workspace\Module2.1_Test\target\dependency-check-report.html#l15_3734223040040e8c3fecd5faa3ae8a1ed6da146b)

[**CVE-2020-5421**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-5421)

According to NVD, CVE-2020-5421 details: “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.”

Solution is to update Spring Framework to 5.2.9 or the following:

5.1.18

5.0.19

4.3.29

[hibernate-validator-6.0.18.Final.jar](file:///C:\Users\Vic\eclipse-workspace\Module2.1_Test\target\dependency-check-report.html#l2_7fd00bcd87e14b6ba66279282ef15efa30dd2492)

[**CVE-2020-10693**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-10693)

CVE-2020-10693: A flaw found in Hibernate Validator ; “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.”

Only known solution is to upgrade affected systems libhibernate-validator-java package in distro.