Software Dependability Analysis of Apache's Commons-CSV Library

Yash Banke Bihari Mittal*

University of Salerno, Fisciano, Italy, y.mittal@studenti.unisa.it

CCS CONCEPTS • Software and its engineering → Software testing and debugging; Software maintenance tools; Open source model.

Additional Keywords and Phrases: Java, Apache Commons CSV, Apache Commons, Maven, Automated Test Case Generation, Bug Fixing, Code Coverage, Code Quality, Snyk, EvoSuite, FindSecBugs, JaCoCo, OWASP DC, PiTest, Software Analytics, Software Dependability, Software Testing, Software Vulnerabilities, SonarCloud.

ABSTRACT

This analysis comprehensively assessed the dependability of the Apache Commons CSV library using various tools and approaches. SonarCloud revealed good code quality and maintainability, while Jacoco reported near-perfect code coverage. PiTest identified areas for improvement through mutation testing, but performance was already deemed satisfactory. EvoSuite generated additional test cases, and security analyses with FindSecBugs, OWASP Dependency Check, and Snyk found no major vulnerabilities after updating dependencies. Overall, this well-maintained, well-tested, and secure library is suitable for working with CSV data in Java.

INTRODUCTION

Core Functionality:

- Offers robust and efficient methods for reading and writing CSV data.
- Supports various CSV dialects and configurations, including custom separators, escape characters, and quoting styles.
- Provides flexible options for processing CSV data, including handling empty lines, comments, and headers.
- Integrates seamlessly with other Apache Commons libraries like BeanUtils and IO for further data manipulation.

Key Features:

- Flexibility: Handles diverse CSV formats and provides customization options.
- Efficiency: Optimized for high performance when dealing with large datasets.
- Ease of Use: Offers a clear and well-documented API for developers.
- Open-Source: Continuously maintained and improved by the community.

Applications:

- Data Import/Export: Widely used for transferring data between applications and formats.
- Data Analysis: Efficiently parses and processes CSV data for analysis.
- Configuration Management: Reads and writes configuration files in CSV format.
- Logging: Captures and analyzes log data stored in CSV files.

LINK TO THE REPOSITORY

https://github.com/ymittalunisa/commons-csv

*

1 SOFTWARE QUALITY ANALYSIS USING SONAR CLOUD

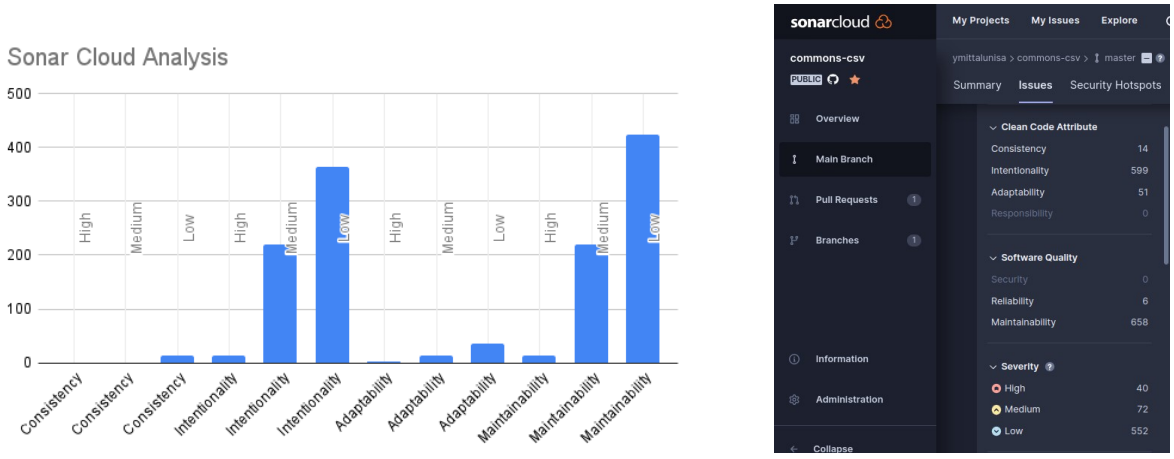


Figure 1.1: Analysis of the master repository

Fixes Applied to Repository (Issues were solved considering the severity):

1.Improved Cognitive Complexity:

a) createConverter: Reduced from 18 to 15 by extracting helper methods and breaking down logic.

b) validate: Reduced from 25 to 15 by extracting helper methods and simplifying checks.

c) createHeaders: Reduced from 25 to 15 by separating parsing and mapping, handling specific cases.

d) read: Reduced from 17 to 15 by improving readability and modularity of reading logic.

e) nextToken: Reduced from 27 to 15 by breaking down logic and simplifying token determination.

f) parseEncapsulatedToken: Reduced from 23 to 15 by extracting logic and simplifying checks.

g) parseSimpleToken: Reduced from 20 to 15 by extracting logic and simplifying token type assignment.

h) checkQuoteCondition: Reduced from 24/31 to 15 by extracting logic, simplifying checks, and combining conditions.

i) Assertions were added to validate test cases.

2. Other Fixes:

a) Defined a constant instead of duplicating "format" literal.

b) Avoided throwing same checked exception multiple times.

c) Changed test method to non-public within the package.

3. Ignored Issue which were further causing issues:

a) 8 Naming convention problems related to an enumerator caused an error.

b) 2 code smells that were relating to adding test case assertions.

c) 3 methods with cognitive complexity of 45,47 and 37.

d) 2 False positives: methods returning ‘null’ values whose return type if changed would hinder the semantics of the code.

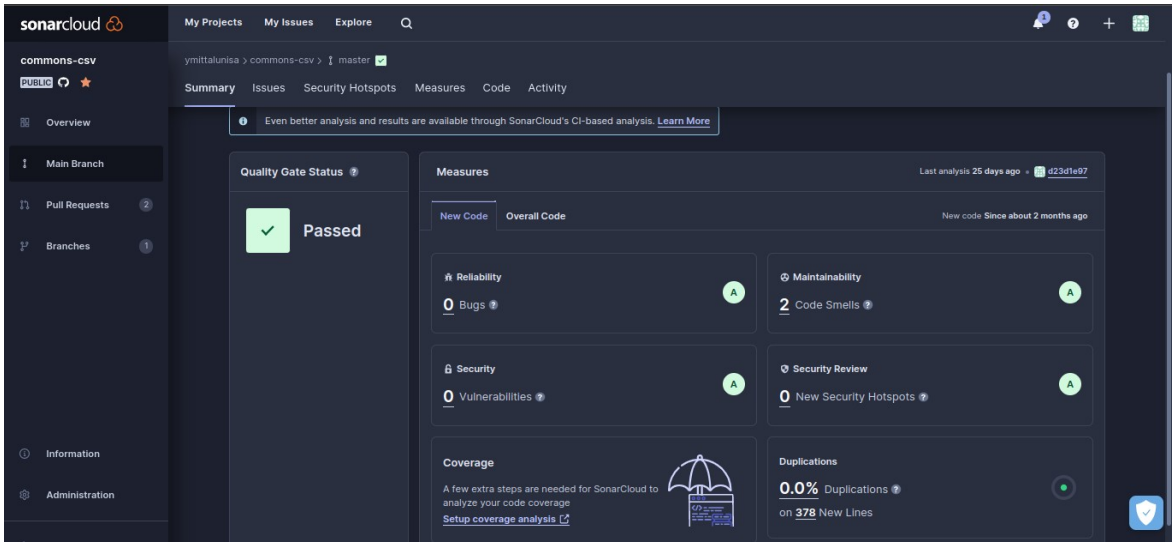


Figure 1.2: Screenshot after applying fixes

2 CODE COVERAGE COMPUTATION USING JACOCO

The code coverage for the repository is currently 98%, which is considered to be very good in many industries. While it is possible to achieve higher coverage, the effort and risks required to do so would likely be significant.

Commands used:

$ mvn clean && mvn verify && mvn jacoco:report

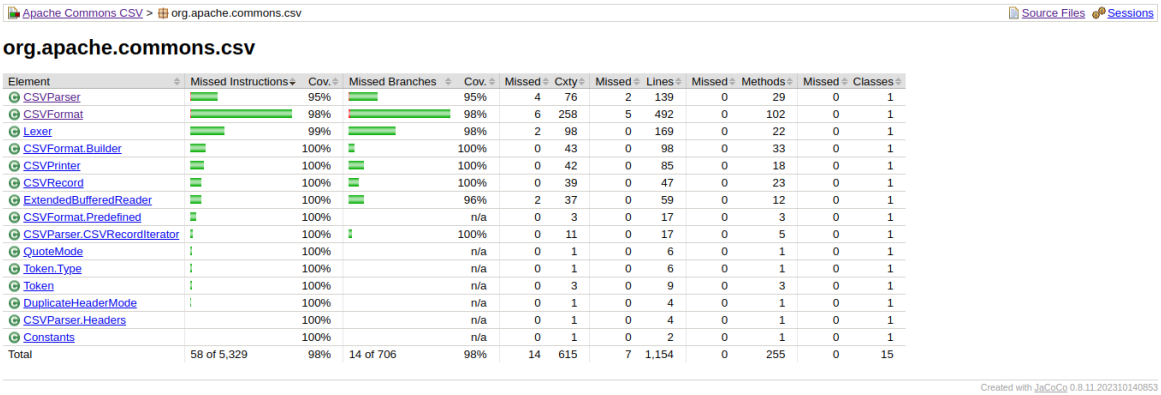


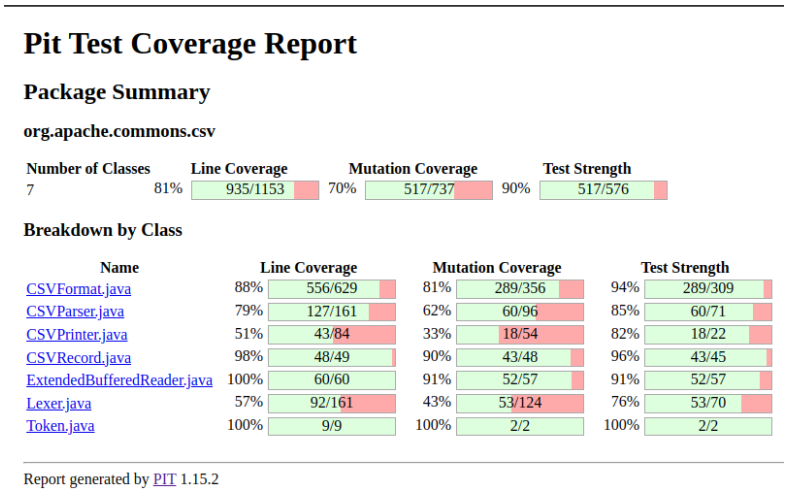
Figure 2: Jacoco code coverage report

3 MUTATION TESTING USING PITEST

Commands used:

$ mvn test-compile org.pitest:pitest-maven:mutationCoverage

$ mvn -DwithHistory test-compile org.pitest:pitest-maven:mutationCoverage



Pit Test Coverage Report

Package Summary

org.apache.commons.csv

Number of Classes	Line Coverage		Mutation Coverage		Test Strength	
7	81%	935/1153	70%	517/737	90%	517/576

Breakdown by Class

Name	Line Coverage		Mutation Coverage		Test Strength	
CSVFormat.java	88%	556/629	81%	289/356	94%	289/309
CSVParser.java	79%	127/161	62%	60/96	85%	60/71
CSVPrinter.java	51%	43/84	33%	18/54	82%	18/22
CSVRecord.java	98%	48/49	90%	43/48	96%	43/45
ExtendedBufferedReader.java	100%	60/60	91%	52/57	91%	52/57
Lexer.java	57%	92/161	43%	53/124	76%	53/70
Token.java	100%	9/9	100%	2/2	100%	2/2

Report generated by PIT 1.15.2

Fig 3: PiTest Coverage Report

4 PERFORMANCE TEST

The Apache Commons CSV library already includes a comprehensive performance test suite(PerformaceTest.java) that covers a variety of different parsing scenarios. The results of the tests show that the performance of the Commons CSV parser is good.

```
[INFO]
[INFO] -------------------------------------------------------
[INFO]  T E S T S
[INFO] -------------------------------------------------------
[INFO] Running org.apache.commons.csv.perf.PerformanceTest
Found test fixture /tmp/worldcitiespop.txt: 132,739,327 bytes.
File parsed in 9,672 milliseconds with Commons CSV: 2,797,246 lines.
File parsed in 7,945 milliseconds with Commons CSV: 2,797,246 lines.
File parsed in 7,714 milliseconds with Commons CSV: 2,797,246 lines.
File parsed in 7,772 milliseconds with Commons CSV: 2,797,246 lines.
File parsed in 7,740 milliseconds with Commons CSV: 2,797,246 lines.
File parsed in 7,700 milliseconds with Commons CSV: 2,797,246 lines.
File parsed in 7,732 milliseconds with Commons CSV: 2,797,246 lines.
File parsed in 8,189 milliseconds with Commons CSV: 2,797,246 lines.
File parsed in 8,191 milliseconds with Commons CSV: 2,797,246 lines.
File parsed in 8,058 milliseconds with Commons CSV: 2,797,246 lines.
Best time out of 10 is 7,700 milliseconds.
File read in 825 milliseconds: 2,797,246 lines.
File read in 722 milliseconds: 2,797,246 lines.
File read in 713 milliseconds: 2,797,246 lines.
File read in 757 milliseconds: 2,797,246 lines.
File read in 710 milliseconds: 2,797,246 lines.
File read in 707 milliseconds: 2,797,246 lines.
File read in 704 milliseconds: 2,797,246 lines.
File read in 711 milliseconds: 2,797,246 lines.
File read in 724 milliseconds: 2,797,246 lines.
File read in 711 milliseconds: 2,797,246 lines.
Best time out of 10 is 704 milliseconds.
[INFO] Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 88.34 s -- in org.apache.commons.csv.perf.PerformanceTest
```

Fig 4: Output of PerformanceTest.java

5 AUTOMATED TEST CASE GENERATION USING EVOSUITE

Commands used:

$ mvn compile

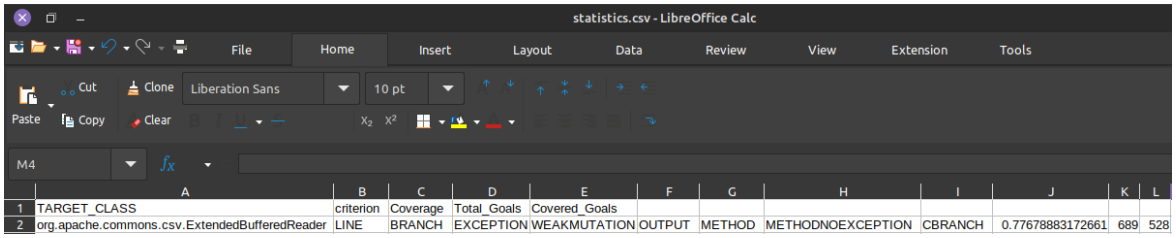
$ mvn evosuite:generate \

-Dclass=org.apache.commons.ExtendedBufferedReader

$ mvn evosuite:info

$ mvn evosuite:export

$ java -jar evosuite-1.0.6.jar -class ExtendedBufferedReader -criterion LINE, BRANCH, EXCEPTION,WEAKMUTATION,OUTPUT,METHOD,METHODNOEXCEPTION,CBRANCH -projectCP target/classes



	A	B	C	D	E	F	G	H	I	J	K	L
1	TARGET_CLASS	criterion	Coverage	Total_Goals	Covered_Goals							
2	org.apache.commons.csv.ExtendedBufferedReader	LINE	BRANCH	EXCEPTION	WEAKMUTATION	OUTPUT	METHOD	METHODNOEXCEPTION	CBRANCH	0.77678883172661	689	528

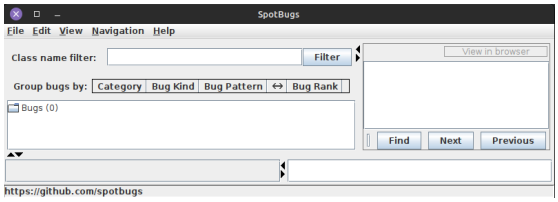
Fig 5.1: statisctics.csv

6 SECURITY ANALYSIS , OWASP DC AND SNYK

FindSecBugs(SpotBugs):-

Commands used:

$ mvn spotbugs:check && mvn spotbugs:gui



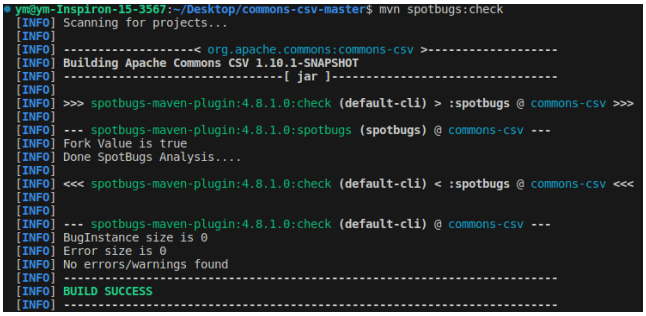


Fig 6.1 Spotbugs Analysis Output in GUI and Terminal

OWASP DC:-

The below .jar files were updated to their latest versions present on maven central as vulnerabilities were found in a few outdated versions of the .jar files:

1. plexus-interpolation
2. plexus-classworlds
3. plexus-component-annotations

4. bcprov-jdk18on
5. maven-core
6. maven-settings
7. maven-shared-utils
8. bcpg-jdk18on

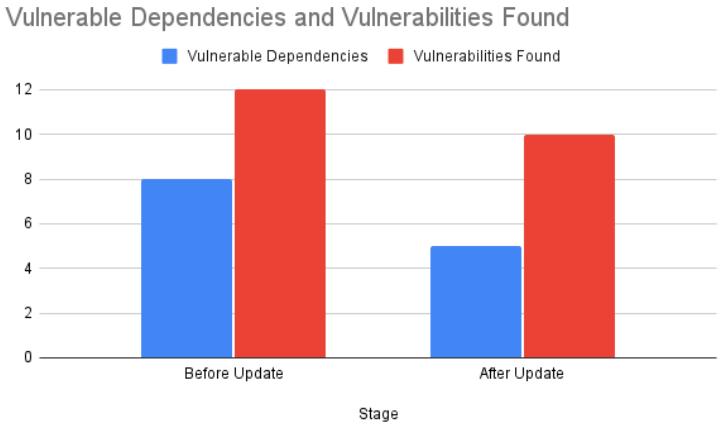
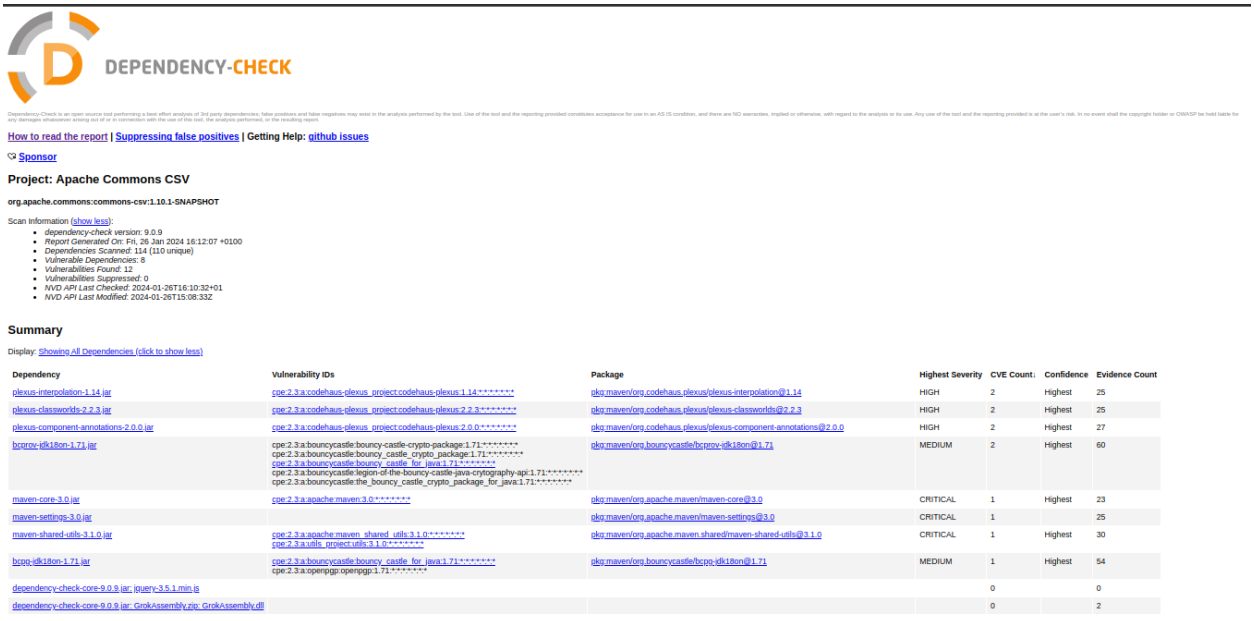


Fig 6.2.1: Number of vulnerabilities analyzed by OWASPDC

Commands used:

$mvn org.owasp:dependency-check-maven:check



DEPENDENCY-CHECK

How to read the report | Suppressing false positives | Getting Help: github issues

Sponsor

Project: Apache Commons CSV

org.apache.commons:commons-csv:1.10.1-SNAPSHOT

Scan Information (show less):

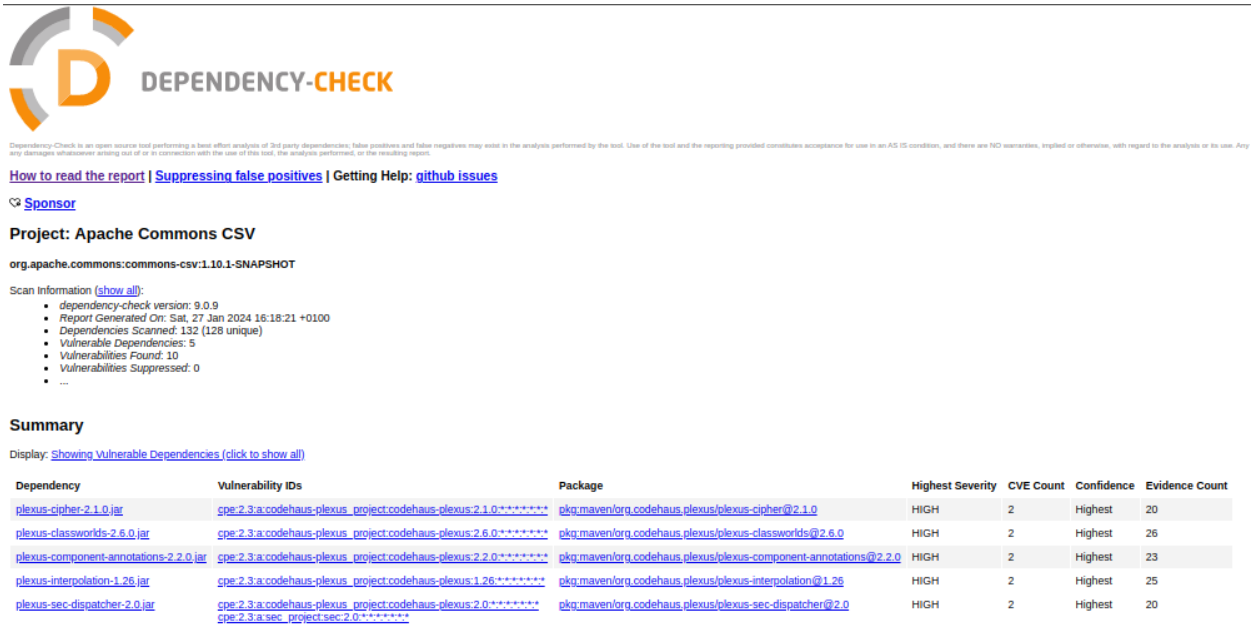
- *dependency-check version:* 9.0.9
- *Report Generated On:* Fri, 26 Jan 2024 16:12:07 +0100
- *Dependencies Scanned:* 114 (110 unique)
- *Vulnerable Dependencies:* 8
- *Vulnerabilities Found:* 12
- *Vulnerabilities Suppressed:* 0
- *NVD API Last Checked:* 2024-01-26T16:10:32+01
- *NVD API Last Modified:* 2024-01-26T15:08:33Z

Summary

Display: Showing All Dependencies (click to show less)

Dependency	Vulnerability IDs	Package	Highest Severity	CVE Count	Confidence	Evidence Count
plexus-interpolation-1.14.jar	cpe:2.3:a:codehaus-plexus_project:codehaus-plexus:1.14:*:*:*:*:*:*:*	pkg:maven/org.codehaus.plexus/plexus-interpolation@1.14	HIGH	2	Highest	25
plexus-classworlds-2.2.3.jar	cpe:2.3:a:codehaus-plexus_project:codehaus-plexus:2.2.3:*:*:*:*:*:*:*	pkg:maven/org.codehaus.plexus/plexus-classworlds@2.2.3	HIGH	2	Highest	25
plexus-component-annotations-2.0.0.jar	cpe:2.3:a:codehaus-plexus_project:codehaus-plexus:2.0.0:*:*:*:*:*:*:*	pkg:maven/org.codehaus.plexus/plexus-component-annotations@2.0.0	HIGH	2	Highest	27
bcprov-jdk18on-1.71.jar	cpe:2.3:a:bouncycastle:bouncy-castle-crypto-package:1.71:*:*:*:*:*:*:* cpe:2.3:a:bouncycastle:bouncy_castle_crypto_package:1.71:*:*:*:*:*:*:* cpe:2.3:a:bouncycastle:bouncy_castle_for_java:1.71:*:*:*:*:*:*:* cpe:2.3:a:bouncycastle:legion-of-the-bouncy-castle-java-cryptography-api:1.71:*:*:*:*:*:*:* cpe:2.3:a:bouncycastle:the_bouncy_castle_crypto_package_for_java:1.71:*:*:*:*:*:*:*	pkg:maven/org.bouncycastle/bcprov-jdk18on@1.71	MEDIUM	2	Highest	60
maven-core-3.0.jar	cpe:2.3:a:apache:maven:3.0:*:*:*:*:*:*:*	pkg:maven/org.apache.maven/maven-core@3.0	CRITICAL	1	Highest	23
maven-settings-3.0.jar		pkg:maven/org.apache.maven/maven-settings@3.0	CRITICAL	1		25
maven-shared-utils-3.1.0.jar	cpe:2.3:a:apache:maven_shared_utils:3.1.0:*:*:*:*:*:*:* cpe:2.3:a:utils_project:utils:3.1.0:*:*:*:*:*:*:*	pkg:maven/org.apache.maven.shared/maven-shared-utils@3.1.0	CRITICAL	1	Highest	30
bcpg-jdk18on-1.71.jar	cpe:2.3:a:bouncycastle:bouncy_castle_for_java:1.71:*:*:*:*:*:*:* cpe:2.3:a:openpgp:openpgp:1.71:*:*:*:*:*:*:*	pkg:maven/org.bouncycastle/bcpg-jdk18on@1.71	MEDIUM	1	Highest	54
dependency-check-core-9.0.9.jar: jquery-3.5.1.min.js				0		0
dependency-check-core-9.0.9.jar: GrokAssembly.zip: GrokAssembly.dll				0		2

Fig 6.2.2: OWASPDC output before updating .jar files



DEPENDENCY-CHECK

How to read the report | Suppressing false positives | Getting Help: github issues

Sponsor

Project: Apache Commons CSV

org.apache.commons:commons-csv:1.10.1-SNAPSHOT

Scan Information (show all):

- *dependency-check version*: 9.0.9
- *Report Generated On*: Sat, 27 Jan 2024 16:18:21 +0100
- *Dependencies Scanned*: 132 (128 unique)
- *Vulnerable Dependencies*: 5
- *Vulnerabilities Found*: 10
- *Vulnerabilities Suppressed*: 0
- ...

Summary

Display: Showing Vulnerable Dependencies (click to show all)

Dependency	Vulnerability IDs	Package	Highest Severity	CVE Count	Confidence	Evidence Count
plexus-cipher-2.1.0.jar	cpe:2.3:a:codehaus-plexus_project:codehaus-plexus:2.1.0:*:*:*:*:*:*:*	pkg:maven/org.codehaus.plexus/plexus-cipher@2.1.0	HIGH	2	Highest	20
plexus-classworlds-2.6.0.jar	cpe:2.3:a:codehaus-plexus_project:codehaus-plexus:2.6.0:*:*:*:*:*:*:*	pkg:maven/org.codehaus.plexus/plexus-classworlds@2.6.0	HIGH	2	Highest	26
plexus-component-annotations-2.2.0.jar	cpe:2.3:a:codehaus-plexus_project:codehaus-plexus:2.2.0:*:*:*:*:*:*:*	pkg:maven/org.codehaus.plexus/plexus-component-annotations@2.2.0	HIGH	2	Highest	23
plexus-interpolation-1.26.jar	cpe:2.3:a:codehaus-plexus_project:codehaus-plexus:1.26:*:*:*:*:*:*:*	pkg:maven/org.codehaus.plexus/plexus-interpolation@1.26	HIGH	2	Highest	25
plexus-sec-dispatcher-2.0.jar	cpe:2.3:a:codehaus-plexus_project:codehaus-plexus:2.0:*:*:*:*:*:*:* cpe:2.3:a:sec_project:sec:2.0:*:*:*:*:*:*:*	pkg:maven/org.codehaus.plexus/plexus-sec-dispatcher@2.0	HIGH	2	Highest	20

Fig 6.2.3: OWASPDC output after updating .jar files

Snyk

No vulnerabilities were found by snyk.

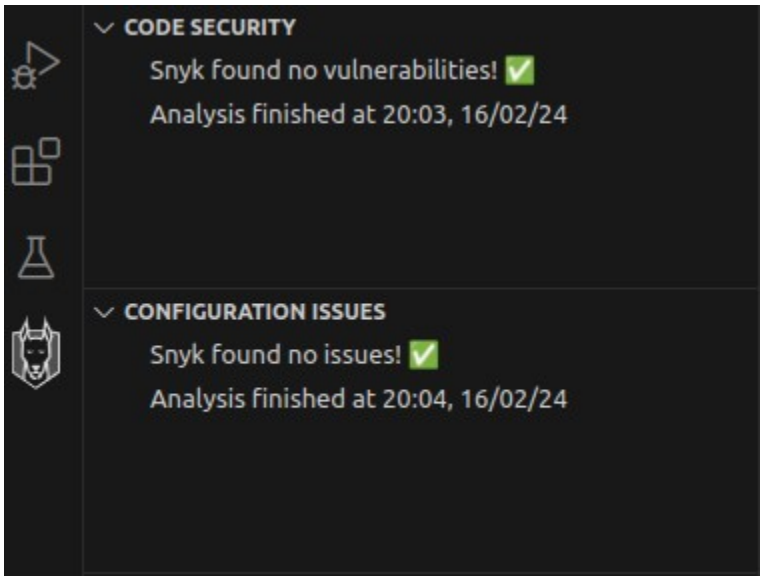


Fig 6.3: Snyk Output

ACKNOWLEDGMENT

I would like to thank Prof. Dario Di Nucci for giving me the opportunity to conduct the above analysis. I would also like to thank Abdul Wasif, Mohammed Aziz, Franco Merenda and my seniors for always supporting and motivating me.

REFERENCES

[1] Baeldung, https://www.baeldung.com
[2] Github, https://github.com
[3] JaCoCo, https://www.jacoco.org/jacoco/
[4] Cobertura MojoHaus, https://www.mojohaus.org/plugins.html
[5] PiTest, https://pitest.org/
[6] PiTest plugin for JUnit5, https://github.com/pitest/pitest-junit5-plugin
[7] Compute code coverage in CI/CD, https://docs.codecov.com/docs/githubtutorial
[8] Tutorials,https://github.com/emaiannone/tools-tutorial/tree/master/evosuite