FINAL REPORT: Characterizing License Practices in Maven Central and Their Relationship with CVEs

Haonan Zhang
haonan.zhang@uwaterloo.ca
University of Waterloo
Waterloo, Canada

Christina Li
christina.li1@uwaterloo.ca
University of Waterloo
Waterloo, Canada

Paul Wooseok Lee
w69lee@uwaterloo.ca
University of Waterloo
Waterloo, Canada

Abstract

Open-source licensing plays an important role in both software reuse and risk management, yet the relationship between license type and security vulnerabilities remains barely explored. In this study, we fill this research gap by systematically analyzing Maven Central artifacts to investigate whether licensing choices correlate with the incidence of Common Vulnerabilities and Exposures (CVEs). Our analysis reveals that while permissive licenses such as Apache-2.0 and MIT are widely adopted and associated with a higher number of vulnerabilities, restrictive licenses like GPL-3.0 and LGPL tend to report fewer CVEs. Furthermore, the Mann-Whitney U test confirms that the difference in CVE incidence between permissive and restrictive licenses is statistically significant. These findings suggest that licensing decisions may have broader implications for software security, thereby providing valuable insights for developers and stakeholders in selecting appropriate licensing strategies.

1 Introduction

Open-source software has been widely used in modern software ecosystems, facilitating the development of various applications, from experimental prototypes to mission-critical organizational platforms [6]. Platforms like Maven Central [9], one of the largest repositories of open-source artifacts, serve as centralized hubs for hosting, managing, and distributing these resources. Most Maven-hosted projects include licenses that provide legal terms and conditions for permissible use, modification, and redistribution, ensuring compliance and clarifying obligations for the users [1]. Apart from licensing, the security of these artifacts is also critical, as vulnerabilities in widely used dependencies can pose systemic risks. To facilitate the address of these risks, Common Vulnerabilities and Exposures (CVEs) are frequently reported by users, promoting stakeholders to identify and fix security flaws in a timely manner [12].

Although there have been many studies regarding the vulnerability and license of the artifacts in Maven Central, most of them either focus only on characterizing and addressing the CVEs [3, 15–17] or only on the license adoption and compliance [4, 11, 13, 14]. To the best of our knowledge, there is a lack of studies on the relationships between the licenses adopted by artifacts and their associated CVEs. The variety of licenses not only determines how code can be modified and redistributed but can also influence the community and governance around a project. Some licenses may encourage broader collaboration or more efficient vulnerability reporting processes, while others may present certain barriers to rapid patching and distribution. This raises the hypothesis that license usage patterns could correlate with a project's vulnerability. To fill this gap, we systematically analyze license information and CVE data associated with Maven Central packages, aiming to shed light on whether certain license types are statistically linked to higher or lower incidences of vulnerabilities. Uncovering such patterns can guide developers and stakeholders to make more informed decisions about artifact and license adoption and keep a balance between permissiveness and security.

To understand the relationships between the license usages and CVE characteristics, we formulate and address the following research questions:

RQ1: What are the characteristics and trends of license adoption and CVE incidence across Maven Central artifacts? In this research question, we aim to investigate the patterns and trends in how licenses are adopted and how security vulnerabilities (CVEs) manifest within Maven Central artifacts. By analyzing historical and current data, we seek to characterize the prevalence and distribution of different license types as well as the frequency and severity of reported CVEs.

RQ2: Do specific license types correlate with higher or lower vulnerability incidence in Maven Central artifacts? In this research question, we investigate whether choosing a permissive or restrictive license is statistically associated with the incidence of CVEs identified, therefore influencing the security of the software.

To support our analysis, we categorize open-source licenses based on how they regulate reuse and redistribution. Permissive licenses (e.g., Apache-2.0, MIT, BSD) allow for broad reuse with minimal conditions, often promoting widespread adoption. Copyleft licenses, such as the GPL, enforce strict requirements that derivative works remain under the same license. Weak copyleft licenses (e.g., LGPL, EPL) apply similar principles, but with more limited scope—typically only when code is directly modified or linked. While these are conceptually distinct, for the purposes of our analysis, we group both strong and weak copyleft licenses under the broader category of "restrictive licenses." This grouping allows for a clearer comparison with permissive licenses in relation to security vulnerabilities.

We make the following contributions with this paper:

- We present the first systematic analysis of the relationship between open-source license types and reported security vulnerabilities (CVEs) in Maven Central.
- We collect and analyze license and CVE data from 900 widely used artifacts, selected across the top 9 license types, using Libraries.io and the Weaver API.
- We apply statistical testing (Mann-Whitney U test) and find a statistically significant difference in normalized CVE rates between permissive and restrictive licenses, supporting our hypothesis that license type correlates with vulnerability incidence.
- We discuss the challenges posed by multi-licensed artifacts and the unique characteristics of emerging licenses like MulanPSL-2.0, outlining opportunities for future exploration into their impact on software security.

2 Methodology

As Figure 1 illustrates, we first extracted all library names from the Neo4j graph database. We then developed a Python script that uses these names to query Libraries.io for license and popularity data. The script includes a retry loop to handle temporary network issues and rate limit problems, ensuring that only valid responses are processed. Once a valid response is received, unnecessary details such as version information are removed to simplify later analysis. To speed up the process of collecting data from the large dataset, we used multi-threading to fetch and store data concurrently. Due to time constraints, we collected license information for 278,984 artifacts over about three days, rather than waiting for data on approximately 600,000 artifacts to be collected.

After the license information was collected, we then collected the CVEs information of the related artifacts. As collecting the CVE information for all of the artifacts we collected earlier can take a significant time, we decided to set our focus on the most important license and artifacts. Specifically, from the license information for 278,984 artifacts, we identified the top 10 most commonly used licenses. We found that the top 9 licenses constitute the majority of the license types used by the artifacts we collected, while the rest of the licenses only constitute a minor portion of the artifacts we collected, hence, we decided to set our focus on these 9 commonly used licenses. For each license, we selected the top 100 most popular artifacts based on the dependents count information we collected from Libraries.io. The dependents count refers to the number of artifacts that are using this artifact and can be a reliable indicator of its popularity. After the most important libraries were identified, we collected the CVE information for these artifacts. In particular, we collected the information about the CVEs reported in the past 5 years as well as the number of the releases in this period.

3 Analysis and Findings

This section presents our key findings based on the two research questions. We analyze trends in license adoption across Maven Central and examine how these trends relate to the occurrence and severity of Common Vulnerabilities and Exposures (CVEs).

3.1 What are the characteristics and trends of license adoption and CVE incidence across Maven Central artifacts?

To address RQ1, we analyze the distribution and characteristics of the most common license types in Maven Central, along with their associated CVE patterns. We also account for the fact that some libraries may be distributed under multiple licenses, which can affect how their security implications are interpreted.

Table 1. Number of releases of 100 top libraries per license

License Name	Total Releases
apache-2.0	30,822
bsd-3-clause	4,276
epl-1.0	8,743
epl-2.0	6,143
gpl-3.0	6,893
lgpl-2.1+	5,405
lgpl-3.0	3,764
mit	15,011
mulanpsl-2.0	4,050

Table 2. Updated number of releases after removing libraries with both permissive and restrictive licenses

License Name	Release Number
apache-2.0	30,822
bsd-3-clause	4,276
epl-1.0	8,628
epl-2.0	1,546
gpl-3.0	5,981
lgpl-2.1+	5,311
lgpl-3.0	3,198
mit	15,011
mulanpsl-2.0	4,050

Characteristics of the licenses. Figure 2 presents the distribution of the top 10 most common licenses identified in our dataset, selected based on dependent count as detailed in Section 2. The figure highlights the dominance of the Apache-2.0 license, which accounts for approximately 68%

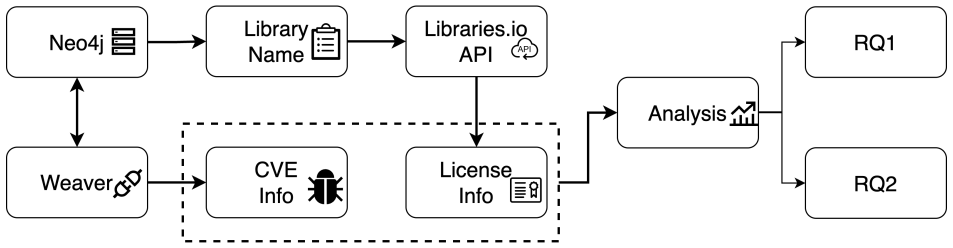

Figure 1. An overview of our approach

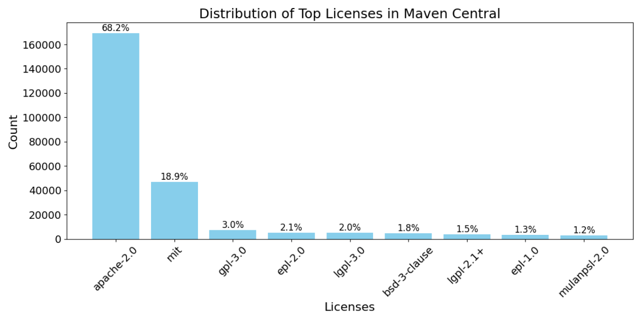

Figure 2. An overview of the distribution of the top licenses in maven central.

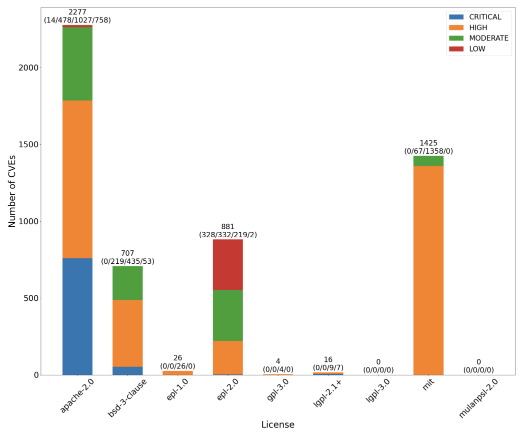

Figure 3. An overview of the distribution of the CVEs and their severity per license.

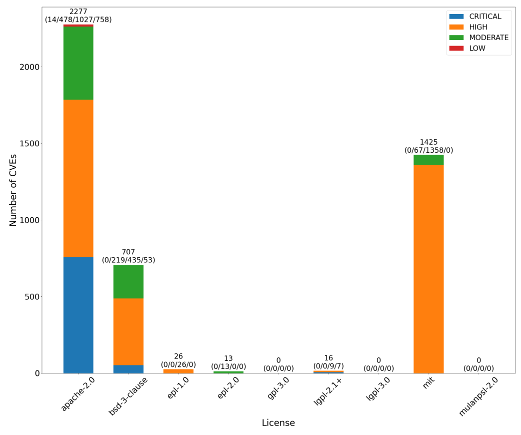

Figure 4. An updated overview of the distribution of the CVEs and their severity per license after removing libraries with both permissive and restrictive licenses.

of all Maven Central artifacts, followed by the MIT license at around 19%. Other commonly observed licenses include BSD-3-Clause (1.8%), GPL-3.0 (3.0%), EPL-2.0 (2.1%), LGPL-3.0 (2.0%), LGPL-2.1+ (1.5%), EPL-1.0 (1.3%), and MulanPSL-2.0 (1.2%). Grouping these by license type, we find that permissive licenses—such as Apache-2.0, MIT, BSD-3-Clause, and MulanPSL-2.0—dominate the ecosystem. Weak copyleft licenses like LGPL and EPL variants and strong copyleft licenses such as GPL-3.0 appear significantly less frequently. These results suggest a strong preference among developers for permissive licensing strategies that may facilitate broader reuse, integration, and adoption.

Characteristics of the CVEs. Characteristics of the CVEs. Table 1 summarizes the number of releases associated with the top 100 libraries for each license type. These releases form the basis for extracting CVEs, providing the data for our vulnerability analysis. Specifically, Figure 3 illustrates the distribution of CVEs categorized by severity levels (CRITICAL, HIGH, MODERATE, and LOW) directly derived from these releases. The artifacts licensed under Apache-2.0 exhibit a notably high number of vulnerabilities, likely due to its extensive adoption. Most CVEs associated with Apache-2.0 fall into the HIGH and CRITICAL severity categories. While the MIT license has also been adopted extensively, its associated CVEs predominantly fall into the HIGH severity category. Interestingly, certain licenses such as LGPL-3.0 and MulanPSL-2.0 report no CVEs, which could indicate either more rigorous security practices or differences in adoption and reporting mechanisms. Restrictive licenses (GPL-3.0, EPL-1.0, and LGPL-2.1+) are generally associated with significantly fewer vulnerabilities, potentially due to their lower adoption rates or more rigorous code maintenance practices.

Potential Causes of the Absence of CVEs for MulanPSL-2.0. One possible reason for the absence of CVEs in MulanPSL 2.0–licensed projects is their being new. MulanPSL 2.0 is a newer license, introduced in January 2020[2], so projects adopting it have had less time to accumulate vulnerability reports. Additionally, these projects are primarily developed and maintained within a different open-source community, where security issues may be reported and tracked a different way and the data are stored in a different database. As a result, even if vulnerabilities exist, they may not be captured in the datasets used in our analysis.

Another factor is the difference in exposure and reporting practices across regions. Permissively licensed projects such as those under Apache 2.0 or MIT, are more prevalent worldwide and are subject to extensive external scrutiny, leading to a higher number of reported vulnerabilities. In contrast, Mulan PSL 2.0–licensed projects may be less visible to international security researchers, and language or regional barriers could result in the under-reporting of vulnerabilities. Therefore, the zero CVE count for Mulan PSL 2.0 does not necessarily imply superior security but likely reflects the combination of limited maturity and differences in global reporting practices.

Impact of Libraries with Multiple Licenses. Unlike other restrictive licenses, EPL-2.0 had unexpectedly high CVE counts. Upon further investigation, we identified that many libraries utilize multiple licenses simultaneously, often mixing permissive and restrictive terms. This mixture can dilute or negate the restrictive characteristics, potentially influencing vulnerability incidence. For instance, most EPL-2.0 libraries also used permissive licenses like Apache-2.0, which conflict with the restrictive intentions of EPL-2.0.

To ensure accuracy in assessing the relationship between licenses and CVEs, we removed libraries that had both permissive and restrictive licenses. The results of this removal process significantly impacted CVE distributions. Particularly notable was EPL-2.0, which initially reported 881 CVEs, primarily stemming from libraries that concurrently used the Apache-2.0 license. After removal, the EPL-2.0 CVE count was drastically reduced to 13. Similar adjustments were made for other licenses, as demonstrated in the updated Figure 4 and Table 2.

We ensured that each of the 100 libraries selected per license category (for a total of 900 libraries across nine licenses) was unique, with no duplicates across license types. This likely happened because the top libraries for each license tend to differ. For example, Apache-2.0 is much more popular than a license like EPL-2.0. Because of this difference in popularity, the most used libraries under one license did not appear under another, so there was no overlap.

This refined analysis now aligns better with our initial hypothesis, demonstrating that permissive licenses (Apache-2.0, BSD-3-Clause, MIT) generally exhibit higher CVEs, whereas restrictive licenses present considerably fewer CVEs.

Finding 1: After removing libraries with multiple licenses (mixing permissive and restrictive licenses), the distribution of CVEs across licenses aligns closely with our hypothesis. Permissive licenses consistently exhibit significantly higher vulnerability counts, whereas restrictive licenses have substantially fewer reported vulnerabilities. This adjustment highlights the critical influence of licensing practices on vulnerability incidence, underscoring that combining permissive with restrictive licenses can notably alter the expected security profile of software libraries.

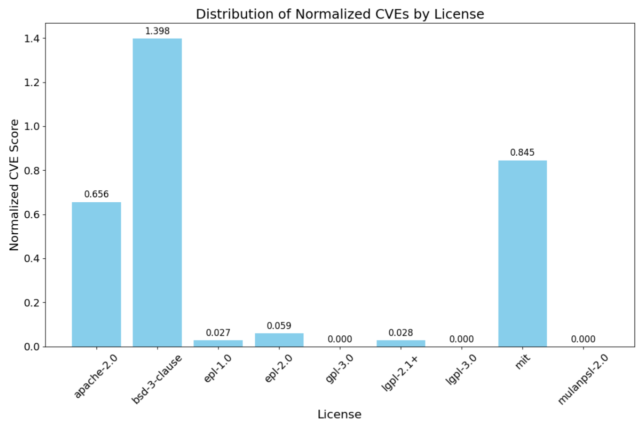

Figure 5. An overview of the distribution of the normalized CVE by license.

3.2 RQ2: Do specific license types correlate with higher or lower vulnerability incidence in Maven Central artifacts?

Figure 5 presents the normalized CVE scores by license. We first assign a numerical value to each severity category based on the Common Vulnerability Scoring System (CVSS) [5]. Table 3 presents the severity level and its related score range of the CVSS. Based on the CVSS, we assign each critical-level CVE a score of 10, each high-level CVE a score of 9, each moderate-level CVE a score of 7, and each low-level CVE a score of 4. Then, for each license, we multiply each severity count by its corresponding severity score and sum them to get a total severity-weighted CVE count. We then divide this sum by the total number of releases of that license, yielding a normalized score. The release number of each license is given in Table 2.

Characteristics of the normalized CVEs. Figure 5 highlights BSD-3-Clause with the highest normalized score (1.398), indicating significant vulnerability incidence relative to its release frequency. Apache-2.0 (0.656) and MIT (0.845) licenses exhibit moderate vulnerability incidence, consistent with

Table 3. CVSS Severity Rating

Metric Value	Severity Level
0-4	Low
4-7	Moderate
7-9	High
9-10	Critical

their permissive nature. Conversely, restrictive licenses such as GPL-3.0 (0.000), LGPL-2.1+ (0.028), LGPL-3.0 (0.000), and the revised EPL-2.0 (0.059) demonstrate notably lower normalized scores, reflecting fewer vulnerabilities relative to the number of releases. Table 2 was important in recalculating these normalized CVE scores by adjusting the total releases after removing libraries with both permissive and restrictive licenses.

Statistical test. We use the *Mann-Whitney U* test [10] to further investigate whether specific license types are related to the incidence of CVEs in Maven Central Artifacts. We choose the *Mann-Whitney U* test because it does not enforce any assumptions about the distribution of analyzed data and applies to a small number of data points. Based on the SPDX License List [8], we first separate the licenses into two categories: *permissive* (e.g., Apache-2.0, BSD-3-Clause, MIT) and *restrictive* (e.g., GPL-3.0, LGPL variants, EPL variants). We removed the MulanSPL-2.0 as it is very new, and no CVE has been reported. Before conducting the *Mann-Whitney U* test, we propose two hypotheses (i.e, *null hypothesis* and *alternative hypothesis*):

$\boldsymbol{H_0}$: There is no statistically significant difference between permissive and restrictive licenses regarding normalized CVE distributions.

$\boldsymbol{H_1}$: There is a statistically significant difference between permissive and restrictive licenses regarding normalized CVE distributions.

The test is executed at the 5% level of significance, which implies that if p-value ≤ 0.05, the H_0 is rejected but H_1 is supported, and vice versa. The resulting p-value from our *Mann-Whitney U* test is 0.036, which is less than the conventional significance level of 0.05. Hence, we do not reject the alternative hypothesis, suggesting that the observed difference in normalized CVE distributions between permissive and restrictive licenses is statistically significant based on our current data. This indicates that the incidence of the CVEs is probably correlated with the license types used by the artifacts.

Finding 2: The permissive licenses exhibit higher CVE incidence, and the *Mann-Whitney U* test further confirms that there is a statistically significant difference between permissive and restrictive licenses.

Implies. This result indicates that artifacts with more permissive licenses are probably altered with less caution, while artifacts with less permissive licenses are probably altered with more caution, as some alteration may violate the rules defined in the licenses. Another reason can be that a permissive license encourages the participants of the development, and CVEs are more actively reported by the users. Nevertheless, this result could also be a result of the fact that the confounding factor, like popularity, is not properly handled; hence, more CVEs are merely a result of the artifacts being more widely used.

4 Related Work

In this section, we discuss prior studies that are related to our work. Overall, these studies can be categorized into two categories: 1) characterizing and addressing the CVEs; and 2) studying the license adoption and compliance.

4.1 Characterizing and addressing the CVEs

As CVEs can expose the systems to severe risks of being attacked, many studies have set their focus on characterizing and addressing the CVEs. Zhang et al. [17] conducted an empirical analysis of persistent vulnerabilities within the Maven ecosystem, introducing "Ranger," an automated solution to restore secure version ranges, highlighting the critical issue of vulnerabilities persisting due to blocked updates by downstream libraries. Wu et al. [16] focused specifically on Maven and examined the threat posed by upstream vulnerabilities to downstream projects within Maven, finding that most downstream projects are not actually reachable by vulnerable functions despite dependency alerts, revealing a high false-positive rate from current Software Composition Analysis (SCA) tools and emphasizing the need for precise vulnerability assessments. To improve vulnerability identification precision, Wu et al. [15] introduced "VFFinder," leveraging large language models (LLMs) to effectively pinpoint vulnerable functions directly from CVE descriptions, significantly reducing false positives in vulnerability reporting by existing SCA tools. Düsing and Hermann [3] analyzed the direct and transitive impacts of vulnerabilities across multiple artifact repositories, including Maven, NuGet, and NPM, concluding that vulnerabilities could propagate significantly through transitive dependency chains, thereby impacting numerous artifacts indirectly and highlighting the complexities developers face in patch management across different ecosystems.

The above studies investigated the challenges of managing and mitigating software vulnerabilities within open-source ecosystems, while our study focuses on disclosing the relationship between the license types of the artifacts and their reported CVEs.

4.2 Studying the license adoption and compliance

Recent research on open-source software licensing has explored various dimensions of licensing practices, challenges, and implications within software ecosystems. Wu et al. [14] conducted a large-scale empirical study across multiple package management platforms, including Maven and NPM, highlighting patterns in license naming, prevalent incompatibility issues, and trends in license evolution, disclosing the complexity and inconsistency in current licensing practices. Vendome et al. [13] investigated the reasons developers adopt or change software licenses, finding that licensing choices are strongly influenced by reuse considerations, commercial integration needs, and inherent biases toward specific licensing attitudes. Pícha and Serbout [11] introduced a pattern-based approach to help developers navigate OSS licensing, presenting structured guidance for selecting appropriate licenses and ensuring compliance through education and enforcement, thereby addressing the confusion developers often face when choosing from the wide variety of licensing options.German et al. [4] proposed methods for auditing and understanding licensing compatibility within software distributions, such as Fedora Linux, uncovering significant inconsistencies and compatibility issues arising from package dependencies and declared licenses, hence advocating for improved automated tools to assist in comprehensive license auditing and compliance management.

The above studies investigated the challenges of selecting, managing, and auditing open-source software licenses, whereas our study specifically examines the relationship between artifact license types and their associated security vulnerabilities (CVEs), offering a different perspective on licensing implications.

5 Threats to Validity & Future Works

5.1 Threats to Validity

Several limitations may affect the validity of our findings. First, the extraction of library metadata and license information proved to be very time-intensive. We managed to extract approximately 278,984 records over a three-day period, even though Libraries.io [7] contains around 685,620 Maven Central records. This reduced sample size may introduce selection bias, and our results may not fully represent the overall ecosystem.

Furthermore, due to time constraints, our analysis was limited to the top 10 most common licenses in the extracted sample (top 10 including "Others", and we removed this one). For each license category, only the top 100 (total of 900) libraries were selected based on the dependents count. Although the dependents count is a useful indicator of a library's popularity, focusing solely on these subsets could overlook important data from libraries that do not meet the inclusion threshold. This selective sampling might restrict the generalisability of the conclusions regarding the link between license type and vulnerability incidence.

Another limitation concerns the temporal restrictions applied to the CVE data. Our study only considered libraries with release timestamps within a specific period—from a point in 2020 to a point in 2025. Such a constraint may not capture the full history of vulnerabilities, particularly for licenses like MulanPSL-2.0, which were introduced in early 2020. As a result, it is unclear whether the observed CVE counts reflect vulnerabilities inherited from earlier versions or those that emerged in later releases.

In addition, the current capabilities of the Weaver API pose a challenge. The API allows for the retrieval of CVE information for a given library, but it does not support the reverse operation—extracting libraries based on CVE data. This unidirectional query approach limits the scope of our analysis and may have prevented us from exploring other aspects of the vulnerability landscape that could be obtained through reverse querying.

5.2 Future Work

Future work should address the above limitations by analysing the full dataset or developing more efficient data extraction methods, perhaps through distributed processing, to capture the full set of available records. Expanding the analysis beyond the top 10 licenses and incorporating multiple ranking metrics beyond the dependents count would also help provide a more comprehensive view. Moreover, extending the temporal range for CVE data and enhancing the API to allow reverse queries from vulnerabilities to libraries would significantly improve the robustness of the study. Such enhancements would allow for a deeper investigation into how license types correlate with the occurrence and severity of vulnerabilities in open-source libraries, which may lead to different conclusions hence provide practical insights into licensing repositories for future developers.

Future studies could delve deeper into the implications of libraries that adopt multiple licenses, particularly those mixing permissive and restrictive types. While this study removed such libraries to preserve license purity in analysis, these "multi-licensed" artifacts represent a complex reality in open-source ecosystems. Understanding how multi-licensed projects handle vulnerability disclosure could reveal new patterns in the relationship between licensing strategies and software security. Additionally, identifying whether certain license combinations are more prone to CVEs than others could inform best practices for multi-license adoption.

Given the complete absence of CVEs associated with MulanPSL -licensed projects in our dataset, future research should investigate whether this is due to superior security practices, limited adoption, or gaps in vulnerability reporting. MulanPSL-2.0 is a relatively new license, introduced in 2020, and it is primarily adopted within Chinese open-source communities. As a result, it is possible that security disclosures are managed through alternative platforms or databases that are not integrated with mainstream CVE tracking systems. A targeted examination of MulanPSL-licensed projects, including qualitative analysis of their development communities, release practices, and regional disclosure norms, could help clarify whether the low CVE count reflects genuinely stronger security or simply under-reporting.

6 Conclusion

This study explores the under-examined relationship between open-source software licensing and security vulnerabilities within the Maven Central ecosystem. By analyzing a curated dataset of libraries and their associated CVEs, we identified clear patterns linking license types to vulnerability incidence. Our findings show that permissive licenses, such as Apache-2.0, MIT, and BSD-3-Clause, are more frequently associated with higher numbers and severity of CVEs. In contrast, restrictive licenses like GPL-3.0, LGPL, and EPL are generally linked to fewer vulnerabilities, even after normalizing for release counts. To validate these observations, we applied the Mann-Whitney U test, which confirmed a statistically significant difference in normalized CVE distributions between permissive and restrictive license groups. This suggests that licensing practices may influence, or at least correlate with, how vulnerabilities are reported and addressed in open-source libraries. While these insights contribute to our understanding of license-security dynamics, we also highlight areas for future research, particularly around multi-licensed projects and less widely adopted licenses like MulanPSL-2.0. By shedding light on these patterns, we aim to help developers, maintainers, and policymakers make more informed decisions when choosing or evaluating licenses for open-source software.

References

[1] Daniel A. Almeida, Gail C. Murphy, Greg Wilson, and Michael Hoye. 2019. Investigating whether and how software developers understand open source software licensing. *Empirical Softw. Engg.* 24, 1 (Feb. 2019), 211–239. https://doi.org/10.1007/s10664-018-9614-9

[2] China Open Source Software License. [n. d.]. Mulan Public Software License, Version 2.0. https://license.coscl.org.cn/MulanPSL2. Accessed: March 30, 2025.

[3] Johannes Düsing and Ben Hermann. 2022. Analyzing the Direct and Transitive Impact of Vulnerabilities onto Different Artifact Repositories. *Digital Threats* 3, 4, Article 38 (Feb. 2022), 25 pages. https://doi.org/10.1145/3472811

[4] Daniel M. German, Massimiliano Di Penta, and Julius Davies. 2010. Understanding and Auditing the Licensing of Open Source Software Distributions. In *Proceedings of the 2010 IEEE 18th International Conference on Program Comprehension (ICPC '10)*. IEEE Computer Society, USA, 84–93. https://doi.org/10.1109/ICPC.2010.48

[5] Henry Howland. 2022. CVSS: Ubiquitous and Broken. *Digital Threats* 4, 1, Article 1 (Feb. 2022), 12 pages. https://doi.org/10.1145/3491263

[6] Xuetao Li, Yuxia Zhang, Cailean Osborne, Minghui Zhou, Zhi Jin, and Hui Liu. 2025. Systematic Literature Review of Commercial Participation in Open Source Software. *ACM Trans. Softw. Eng. Methodol.* 34, 2, Article 33 (Jan. 2025), 31 pages. https://doi.org/10.1145/3690632

[7] Libraries.io. 2023. *Libraries.io: The Open Source Discovery Service.* https://libraries.io/ [Online; accessed 21-February-2025].

[8] SPDX License List. 2025. *The SPDX License List is an integral part of the SPDX Specification.* https://spdx.org/licenses/ [Online; accessed 25-March-2025].

[9] Frederic P. Miller, Agnes F. Vandome, and John McBrewster. 2010. *Apache Maven.* Alpha Press.

[10] Nadim Nachar. 2008. The Mann-Whitney U: A Test for Assessing Whether Two Independent Samples Come from the Same Distribution. *Tutorials in Quantitative Methods for Psychology* 4 (03 2008). https://doi.org/10.20982/tqmp.04.1.p013

[11] Petr Pícha and Souhaila Serbout. 2024. On the Adoption of Open Source Software Licensing - A Pattern Collection. In *Proceedings of the 29th European Conference on Pattern Languages of Programs, People, and Practices (EuroPLoP '24)*. Association for Computing Machinery, New York, NY, USA, Article 19, 7 pages. https://doi.org/10.1145/3698322.3698341

[12] Henrik Plate, Serena Elisa Ponta, and Antonino Sabetta. 2015. Impact assessment for vulnerabilities in open-source software libraries. In *Proceedings of the 2015 IEEE International Conference on Software Maintenance and Evolution (ICSME) (ICSME '15)*. IEEE Computer Society, USA, 411–420. https://doi.org/10.1109/ICSM.2015.7332492

[13] Christopher Vendome, Mario Linares-Vasquez, Gabriele Bavota, Massimiliano Di Penta, Daniel M. German, and Denys Poshyvanyk. 2015. When and why developers adopt and change software licenses. In *Proceedings of the 2015 IEEE International Conference on Software Maintenance and Evolution (ICSME) (ICSME '15)*. IEEE Computer Society, USA, 31–40. https://doi.org/10.1109/ICSM.2015.7332449

[14] Jiaqi Wu, Lingfeng Bao, Xiaohu Yang, Xin Xia, and Xing Hu. 2024. A Large-Scale Empirical Study of Open Source License Usage: Practices and Challenges. In *Proceedings of the 21st International Conference on Mining Software Repositories* (Lisbon, Portugal) *(MSR '24)*. Association for Computing Machinery, New York, NY, USA, 595–606. https://doi.org/10.1145/3643991.3644900

[15] Yulun Wu, Ming Wen, Zeliang Yu, Xiaochen Guo, and Hai Jin. 2024. Effective Vulnerable Function Identification based on CVE Description Empowered by Large Language Models. In *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering* (Sacramento, CA, USA) *(ASE '24)*. Association for Computing Machinery, New York, NY, USA, 393–405. https://doi.org/10.1145/3691620.3695013

[16] Yulun Wu, Zeliang Yu, Ming Wen, Qiang Li, Deqing Zou, and Hai Jin. 2023. Understanding the Threats of Upstream Vulnerabilities to Downstream Projects in the Maven Ecosystem. In *Proceedings of the 45th International Conference on Software Engineering* (Melbourne, Victoria, Australia) *(ICSE '23)*. IEEE Press, 1046–1058. https://doi.org/10.1109/ICSE48619.2023.00095

[17] Lyuye Zhang, Chengwei Liu, Sen Chen, Zhengzi Xu, Lingling Fan, Lida Zhao, Yiran Zhang, and Yang Liu. 2024. Mitigating Persistence of Open-Source Vulnerabilities in Maven Ecosystem. In *Proceedings of the 38th IEEE/ACM International Conference on Automated Software Engineering* (Echternach, Luxembourg) *(ASE '23)*. IEEE Press, 191–203. https://doi.org/10.1109/ASE56229.2023.00058