

Severity Without a Common Scale: A Cross-Ecosystem Census of Machine-Readable Severity Signals in OSV

Cybersecurity Advances • Volume 3 • Issue 1 • 2026 • pp. 36–45

ATS PRESS Research Group

AlphaTech Scholarly Press, United States

Correspondence: Md Sazzad Hossain • md@hossain.tech • ORCID 0009-0009-4948-1179

Received 2026-10-02 • Accepted 2026-10-02 • Published 2026-10-02

© 2026 ATS PRESS Research Group. Published by AlphaTech Scholarly Press under CC BY 4.0.

Abstract

Background: Package-security tools need severity metadata they can parse consistently. The Open Source Vulnerability (OSV) schema supports quantitative severity arrays at record and affected-package levels, while source databases can publish local qualitative labels. Their availability across package ecosystems has not been measured at scale.

Methods: We conducted a preregistered census of OSV ecosystem archives acquired on 1 October 2026. Of 280,270 unique records, 278,534 nonwithdrawn records formed the primary denominator. We separated quantitative items from local labels, validated declared CVSS versions, measured placement and provenance, and ran prespecified restrictions. A deterministic rerun reproduced the semantic-result hash.

Results: Only 30,577 active records (10.98%; 95% Wilson CI, 10.86%–11.09%) contained a formed schema-native quantitative severity item. Another 2,031 (0.73%) had only a local label; 245,926 (88.29%) had neither signal. Coverage ranged from 2.85% for npm to 93.50% for Maven. All 35,182 severity items were top-level CVSS vectors: 26,624 v3 and 8,558 v4. None appeared in affected-package severity arrays or populated the optional source field. All v3 vectors and 8,555 v4 vectors were syntactically concordant. MAL records accounted for 95.93% of records without standard severity. Excluding MAL, coverage rose to 75.20%.

Conclusions: Pooled OSV severity coverage is governed by source composition and hides ecosystem differences. Consumers should preserve missingness, version, and provenance instead of treating absent scores and local labels as equivalent. These findings measure represented metadata, not assessment correctness, exploit likelihood, or ecosystem quality.

Keywords: OSV; CVSS; vulnerability metadata; software supply chain; package ecosystems; data quality

1. Introduction

Severity scores sit at an awkward boundary between description and decision. A Common Vulnerability Scoring System (CVSS) vector encodes characteristics of a vulnerability, but operational tools often reduce that vector to a number or band and use it for queue ordering, service-level targets, and reporting. The distinction matters because severity is not exploit probability, local exposure, or business impact [3–5,10]. It also matters because the score may be unavailable, delayed, produced under a different CVSS version, or attributed to a different assessor [6–9,11].

The Open Source Vulnerability (OSV) format was designed for open-source package and version data [1,2,14]. Its severity array can carry one or more quantitative assessments. In schema version 1.9.1, recognized types include CVSS v2, CVSS v3, CVSS v4, and Ubuntu priority. The same array can appear at record level or within an affected-package entry. An optional source field can identify an assessment from

NVD, a CVE Numbering Authority, the home database, or another named source [1]. The schema also permits database-specific extension objects. In practice, those extensions may include qualitative values such as “high” or “moderate,” but their semantics are controlled by the source rather than by the portable OSV severity definition.

Existing severity research establishes why these distinctions should not be collapsed. Evaluators can disagree when assigning CVSS metrics [6]. Severity rankings can differ across data sources [7], and product context can change an assessment [9]. Longitudinal work on NVD has documented version and source fragmentation [8], while scoring delay can create temporary gaps [11]. Those studies concern the quality, timing, or agreement of assessments. They do not answer a more basic interoperability question: when a consumer downloads OSV ecosystem archives, how often is a severity signal present in the standard field, where is it placed, which declared version does it use, and is its assessment source explicit?

We answer that question with a census of six package ecosystems: PyPI, npm, Maven, Go, crates.io, and RubyGems. We separate three mutually exclusive record states: a schema-native quantitative item, a local qualitative label only, and no represented severity signal. We also validate type-vector syntax, examine record versus affected-package placement, measure explicit provenance, and quantify concentration among source families. The aim is descriptive. We do not rescore vulnerabilities, compare the substantive correctness of assessments, or infer why metadata is absent.

This article makes three contributions. First, it provides a preregistered and reproducible measurement of OSV severity representation over 278,534 active records. Second, it shows how a pooled estimate can be dominated by one advisory family and therefore conceal the coverage experienced in individual ecosystems. Third, it identifies two interoperability facts that are easy to miss in aggregate counts: every observed quantitative item was top-level, and every observed item relied on implicit home-database attribution.

2. Related work

CVSS was created to encode vulnerability characteristics in a standardized vector and derive severity scores [3–5]. The major versions are not interchangeable. Their metric sets, formulas, and intended usage have changed, so a pipeline that strips the version prefix loses information needed for correct parsing. FIRST also distinguishes base severity from threat and environmental considerations, especially in CVSS v4 [5].

Empirical studies have examined the human and institutional variation behind severity metadata. Wunder et al. found substantial evaluator disagreement for selected vulnerability types and reported that practitioners still regard CVSS as useful despite known difficulties [6]. Croft et al. compared severity data across reporting sources and showed that inconsistencies can persist through the vulnerability-reporting lifecycle [7]. Coutinho et al. analyzed product-specific context in CVSS assessments [9]. These results caution against treating syntactically valid vectors as ground truth.

Database structure creates a related problem. Walkowski et al. measured fragmentation across CVSS versions and scoring sources in NVD [8]. Ruohonen studied delays in assigning CVSS information [11]. NVD remains a major standards-based enrichment source [13], but OSV aggregates a broader set of package-ecosystem and advisory feeds. Some OSV records describe malicious packages, ecosystem advisories, or records without a CVE identity. A denominator built from those records therefore differs from a CVE-only denominator.

Open-source dependency studies have focused on detection, assessment, and mitigation workflows [12]. OSV extends those workflows with precise package and affected-version representations [1,2]. Yet a schema’s ability to hold a field does not establish field coverage. Our study focuses on that representation

layer. It complements work on scoring disagreement by determining which portable severity artifacts are present before a consumer can compare or operationalize them.

3. Methods

3.1 Protocol and design

We wrote and froze the protocol before computing severity outcomes. The SHA-256 digest of `PROTOCOL.md` is `cb119327bafd44937334ac775dc883f3770710d0e921c207f8bded4b48e26f5c`. The corpus bytes had previously supported studies of OSV alias connectivity and remediation endpoints, but those studies did not report the severity outcomes defined here. Reuse of the snapshot allowed stable cross-study provenance without reusing the question or analysis.

After the first aggregate run, we filed one amendment. A CSV column for affected-level severity counts disappeared when every value was zero because the counter did not materialize zero-valued keys. Amendment 001 required explicit zero columns and did not change any classification, numerator, denominator, sensitivity, or interpretation rule. Its SHA-256 digest is `3ced13f08516031b3d9aa44c0135a27dcb145366bc3f0f1d82878afaaf48ac2a`.

3.2 Data sources and integrity controls

We used official OSV all.zip archives for PyPI, npm, Maven, Go, crates.io, and RubyGems, acquired on 1 October 2026. A manifest preserves archive URLs, acquisition times, response metadata, byte counts, and SHA-256 hashes. The analysis verified every hash and ZIP cyclic-redundancy check before parsing. All 283,913 JSON members parsed as objects. Exact same-ID copies across ecosystem archives were deduplicated by canonical JSON hash. The result was 280,270 unique records. No record ID had divergent content variants, and no record lacked a valid identifier.

We pinned the OSV schema documentation at Git commit `65280faa676c7a69bb9208d876b2419f88dd7e03`, described as `v1.9.1-1-g65280fa`. We excluded 1,736 withdrawn records from the primary denominator, leaving 278,534 active records.

3.3 Severity representations

A schema-native severity item was an object in `top-level severity[]` or in `affected[].severity[]` for one of the six study ecosystems. A formed item required nonempty string type and score values. We retained the optional string source when present. Array members that failed these structural rules were counted as malformed.

We recognized the schema types `CVSS_V2`, `CVSS_V3`, `CVSS_V4`, and `Ubuntu`. CVSS vectors were parsed with the pinned `cvss` Python package. A `v3` item had to parse and begin with `CVSS:3.0/` or `CVSS:3.1/`; a `v4` item had to parse and begin with `CVSS:4.0/`. A `v2` item had to parse under the CVSS 2.0 grammar. Ubuntu priority had to be one of `negligible`, `low`, `medium`, `high`, or `critical` in lowercase. This test establishes syntactic type-score concordance, not the correctness of an assessment.

We treated source as explicit only when it contained a nonempty string. Its absence is schema-valid and implies attribution to the home database [1]. We therefore report missing explicit provenance rather than calling it malformed.

A local qualitative label was a nonempty string in `record-level database_specific.severity`, `affected-entry ecosystem_specific.severity`, or `affected-entry database_specific.severity`. We normalized Unicode, whitespace, and case. These paths are useful for describing fallback signals, but their values are not governed by the portable severity-array semantics.

3.4 Outcomes and statistical analysis

The primary outcome was the proportion of active, nondivergent unique records with at least one formed schema-native severity item. Secondary outcomes covered mutually exclusive signal tiers, field placement, ecosystem-specific record coverage, affected-entry coverage, declared types, syntactic concordance, explicit provenance, local-label values, and source-family concentration among records without standard severity.

Ecosystem denominators were unique record-ecosystem memberships derived from affected-package entries. A record could contribute to more than one ecosystem. Item counts and record counts were never added. We calculated two-sided Wilson 95% intervals for binary proportions. Because the data are a census of a frozen snapshot, these intervals are descriptive precision summaries rather than sampling-based inference about all future OSV states.

We defined source family as the uppercase prefix before the first hyphen in the record identifier. Concentration was summarized with the Herfindahl-Hirschman index (HHI) and top-family shares. Prespecified restrictions included all records including withdrawn, exclusion of MAL, CVE-linked records, records published through 2025, and non-MAL CVE-linked records.

3.5 Software and reproducibility

We developed extraction, validation, statistics, and archive handling with retained failing and passing tests. Nine tests passed before analysis and again after the reporting repair. Canonical JSON, sorted aggregate outputs, exact input hashes, and deterministic duplicate handling were used throughout. The semantic-result SHA-256 was 6bb404c9e0665083bdaf89d2766f43f4f269c01b67f6c66fd3cd166645453d8a on both the amended primary execution and a complete rerun.

4. Results

4.1 Overall availability

Among 278,534 active records, 30,577 contained a formed schema-native quantitative severity item, for coverage of 10.98% (95% CI, 10.86%–11.09%). Another 2,031 records (0.73%; 95% CI, 0.70%–0.76%) had a local qualitative label but no standard item. The remaining 245,926 records (88.29%; 95% CI, 88.17%–88.41%) had neither represented signal.

Local labels therefore closed only a small part of the standard-field gap. Their occurrence counts were dominated by moderate (11,356), high (10,233), critical (3,883), and low (1,985). These are occurrence counts across extension paths, not unique records or a portable cross-source scale.

4.2 Ecosystem differences

Coverage varied by more than thirtyfold (Table 1; Figure 1). Maven had the highest record coverage at 93.50% (6,590/7,048). It was followed by crates.io at 61.80%, Go at 49.49%, PyPI at 41.75%, RubyGems at 16.17%, and npm at 2.85% (6,530/229,049).

Table 1. Schema-native quantitative severity coverage by ecosystem

Ecosystem	Records with severity	Active record-ecosystem memberships	Coverage	95% CI
Maven	6,590	7,048	93.50%	92.90%–94.05%
crates.io	1,697	2,746	61.80%	59.97%–63.60%
Go	4,585	9,264	49.49%	48.47%–50.51%
PyPI	10,611	25,413	41.75%	41.15%–42.36%

RubyGems	861	5,326	16.17%	15.20%–17.18%
npm	6,530	229,049	2.85%	2.78%–2.92%

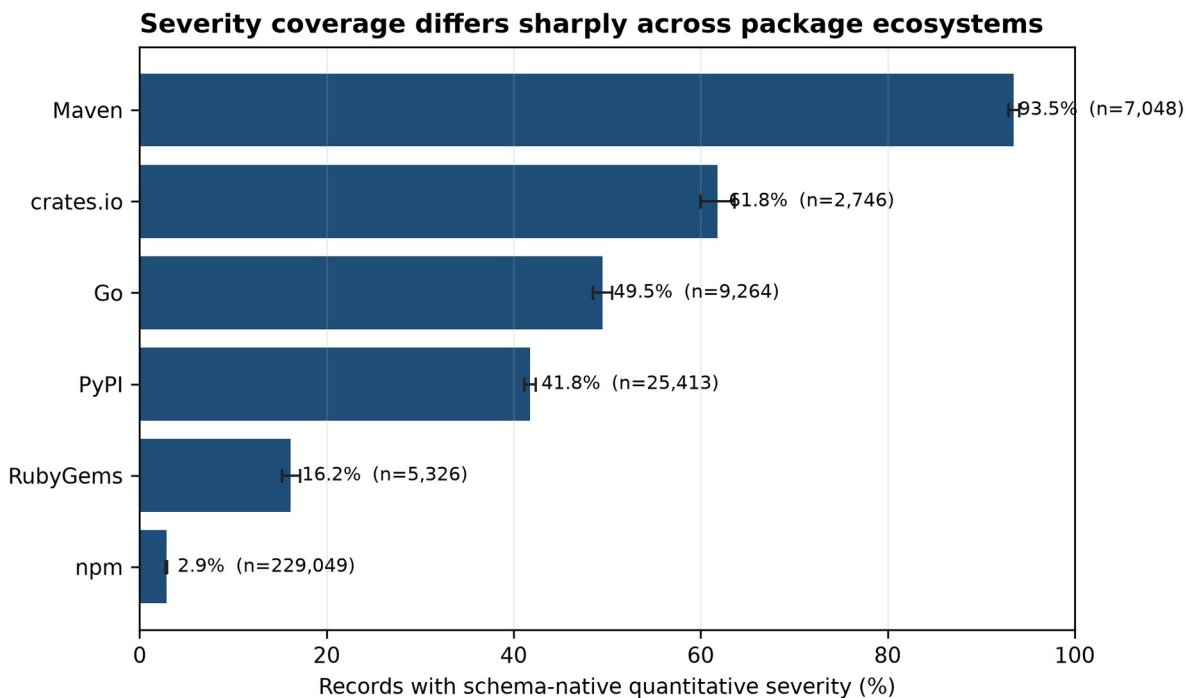


Figure 1. Schema-native quantitative severity coverage by ecosystem. Error bars show Wilson 95% intervals.

Signal tiers showed different fallback patterns (Figure 2). Maven had only two records with neither a standard item nor local label; 456 records (6.47%) were local-only. In npm, 221,829 records (96.85%) had no represented severity signal and 690 (0.30%) were local-only. Local-only coverage was largest in crates.io at 8.34%, but even there it left 29.86% of records with neither signal.

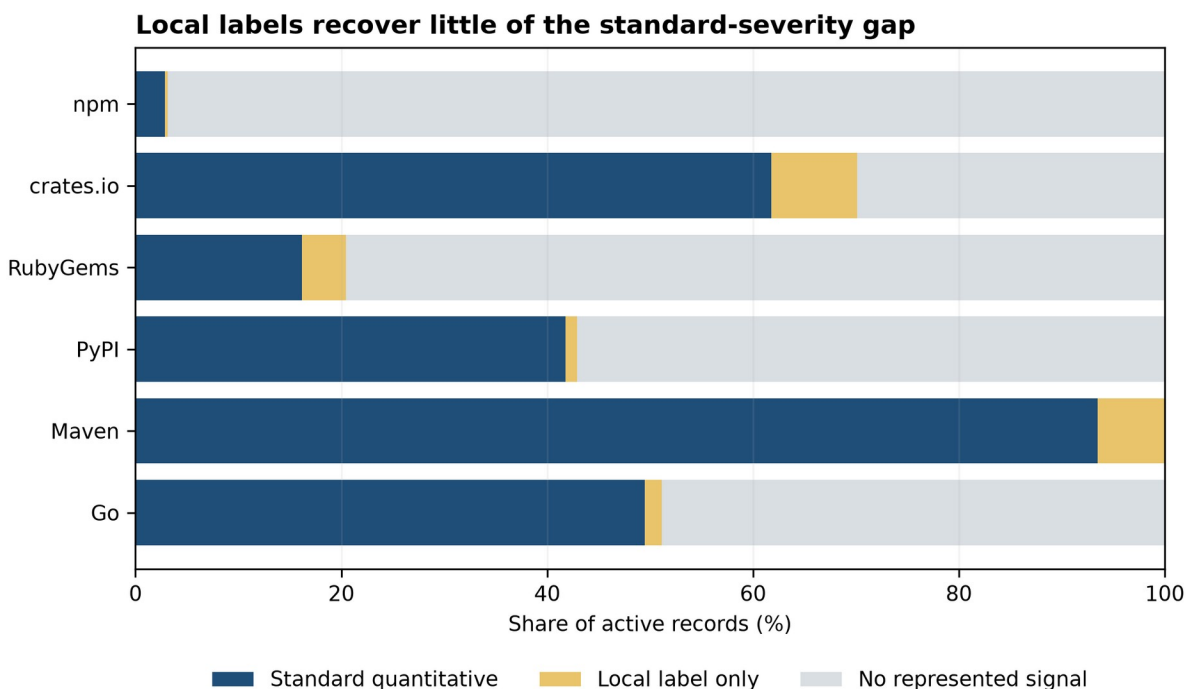


Figure 2. Mutually exclusive severity-signal tiers by ecosystem.

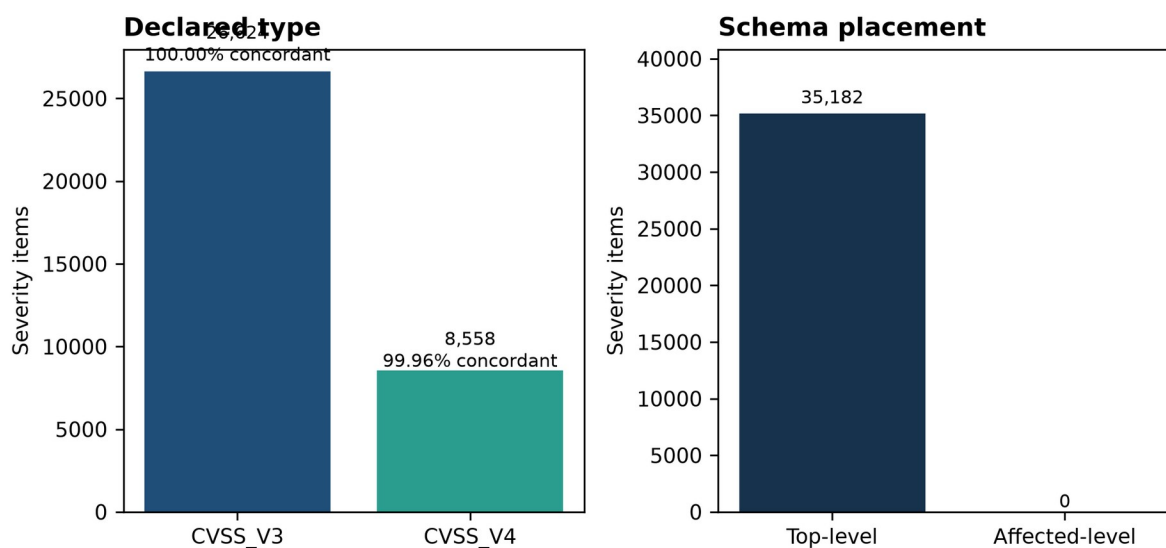
4.3 Type, placement, validity, and provenance

The 30,577 standard-covered records contained 35,182 formed items. Every item appeared at the top level. None of the 301,010 in-scope affected-package entries used `affected[].severity[]`. No malformed severity-array member was observed.

CVSS v3 accounted for 26,624 items (75.68%) and CVSS v4 for 8,558 (24.32%). We observed no formed CVSS v2, Ubuntu, or other declared type. Every v3 vector was syntactically concordant. Of the v4 vectors, 8,555 were concordant and three were not, yielding 99.96% concordance. Figure 3 summarizes type and placement.

None of the 35,182 items populated the optional source field. Under OSV's rules, those items are implicitly attributed to the home database. The absence is not a schema violation, but it prevents consumers from distinguishing explicit NVD, CNA, or self-assessed provenance directly from this field.

All formed items were top-level CVSS vectors with implicit provenance

*Figure 3. Severity item composition, syntactic concordance, and schema placement.*

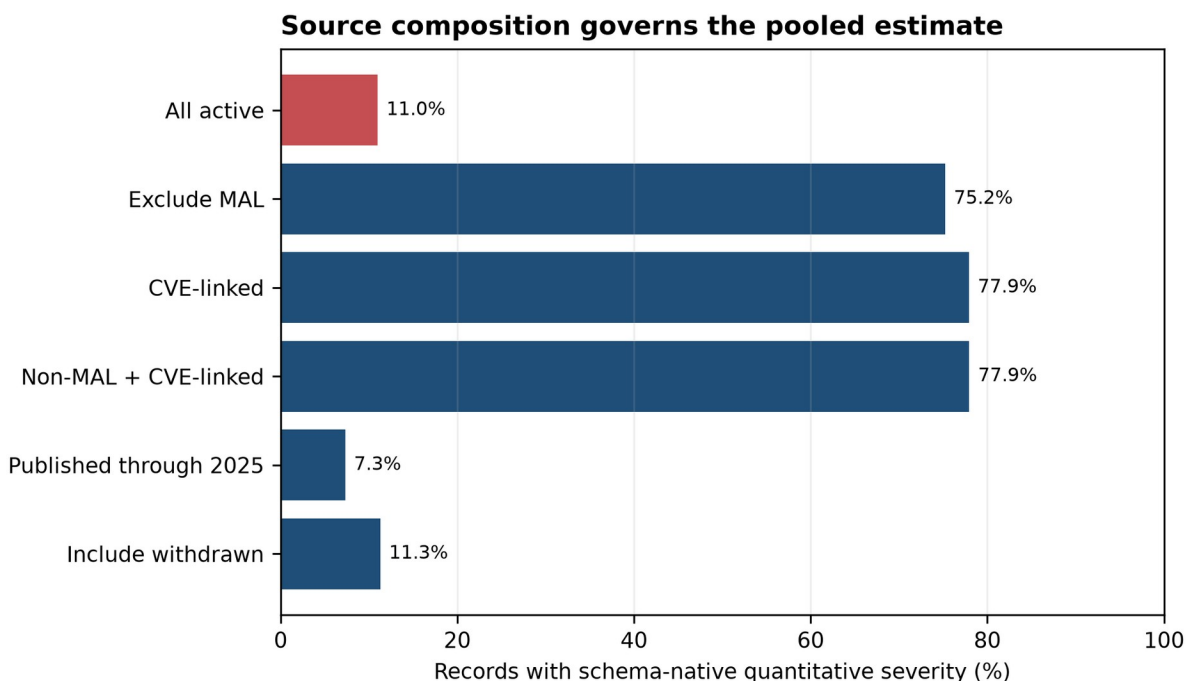
4.4 Source concentration and sensitivity analyses

Records without a standard item were highly concentrated. The MAL family accounted for 237,875 of 247,957 such records (95.93%). GO contributed 4,508 (1.82%), PYSEC 2,738 (1.10%), GHSA 2,024 (0.82%), and RUSTSEC 800 (0.32%). The source-family HHI was 0.921; the top three families accounted for 98.86% of missing-standard records.

This concentration changed the pooled estimate substantially (Table 2; Figure 4). Excluding MAL raised standard coverage from 10.98% to 75.20% (30,577/40,659). Among CVE-linked records, coverage was 77.91% (29,129/37,387). The non-MAL, CVE-linked restriction was similar at 77.95%. Including withdrawn records changed the estimate only modestly, to 11.27%. Records published through 2025 had 7.30% coverage because that restriction retained much of the large historical MAL denominator; it should not be interpreted as a temporal trend from one snapshot.

Table 2. Prespecified sensitivity estimates

Restriction	Records with severity	Denominator	Coverage	95% CI
Active records	30,577	278,534	10.98%	10.86%–11.09%
Include withdrawn	31,595	280,270	11.27%	11.16%–11.39%
Exclude MAL	30,577	40,659	75.20%	74.78%–75.62%
CVE-linked	29,129	37,387	77.91%	77.49%–78.33%
Published through 2025	17,977	246,227	7.30%	7.20%–7.40%
Non-MAL and CVE-linked	29,129	37,370	77.95%	77.52%–78.37%

*Figure 4. Standard severity coverage under prespecified corpus restrictions.*

5. Discussion

5.1 The pooled estimate is a property of corpus composition

The headline estimate, 10.98%, is arithmetically correct but operationally incomplete. npm supplied most record-ecosystem memberships, and MAL records supplied 95.93% of records lacking a standard severity item. A consumer scanning Maven dependencies sees a very different metadata environment from one processing the full archive union. The non-MAL sensitivity of 75.20% is not a “corrected” estimate; it answers a different question. Both estimates are needed to understand the corpus.

Malicious-package advisories also complicate the meaning of absent CVSS. A record can be actionable because the package itself is malicious even when no vulnerability severity vector is attached. Assigning a synthetic score, or treating absence as zero severity, would erase that distinction. Consumers should preserve advisory family and missingness rather than force every record onto one numeric scale.

5.2 Standard fields and local labels are not interchangeable

Only 0.73% of active records were rescued into a local-only tier. Local labels may still be useful within the publishing database, and their values often resemble familiar CVSS bands. Their apparent familiarity does not make them semantically portable. A database-specific high can reflect a source policy, contextual judgment, or derived band that another source would apply differently [6,7,9].

A robust ingestion pipeline should therefore expose at least four fields: the original severity type, original score or label, representation path, and source attribution status. It should not overwrite a schema-native vector with a local label or collapse missing, malformed, and not-applicable states into one null.

5.3 Placement and provenance expose unused schema capacity

OSV permits severity arrays at affected-package level, which could represent package-specific assessments. We observed none across 301,010 in-scope affected entries. Every formed item was top-level. This does not prove that affected-level severity is unnecessary. It shows that a consumer relying only on current six-ecosystem data would not encounter it in this snapshot, while a schema-complete implementation must still support it.

The provenance result is similarly narrow but important. The source field was added to express where an assessment came from [1]. No observed item used it. Implicit home-database attribution is valid, yet it cannot represent the richer provenance distinctions studied in NVD and cross-source research [7,8]. Adoption may increase because the field was recently added to the schema. A single snapshot cannot test that trajectory.

5.4 Valid syntax is necessary but weak evidence

Almost every observed vector passed our version-aware syntax check. This is good news for parsers, but it says little about assessment quality. Human evaluators can choose different metrics for the same vulnerability [6], source rankings can disagree [7], and product context can matter [9]. We did not recalculate scores, compare duplicated vulnerability assessments, or decide which assessor was correct. Our result should be read as transport compatibility, not substantive validation.

5.5 Implications for producers and consumers

For producers, the largest interoperability gain would come from filling the standard severity array when a defensible quantitative assessment exists, while preserving “not assessed” as different from low severity. Producers that import an assessment should use explicit provenance when the schema supports it. Package-specific assessments belong at affected level only when their semantics genuinely differ from the record-level assessment.

For consumers, ecosystem-specific monitoring is safer than one global completeness threshold. Dashboards should report denominator composition, especially advisory families such as MAL. Version-aware parsing is mandatory because v3 and v4 vectors coexist. Pipelines should retain the original vector and should not silently translate qualitative extension labels into CVSS scores.

6. Limitations

This is a cross-sectional census of archives acquired on one date. OSV changes continuously, so the estimates do not describe earlier or later snapshots. The “published through 2025” restriction is not a longitudinal analysis because all records were observed in 2026.

We measured represented fields, not the reasons behind them. Missing severity may reflect advisory type, source policy, migration timing, delayed scoring, unavailable evidence, or a judgment that CVSS is not

appropriate. The concentration analysis groups IDs by prefix; a prefix is descriptive provenance, not proof of one independent assessor or workflow.

Our syntax checks depend on the pinned parser and do not independently implement every CVSS grammar rule. Syntactic concordance does not verify the base score, metric choices, source expertise, or applicability to a user's environment. We also did not normalize local labels across databases because doing so would impose semantics not guaranteed by OSV.

Record-ecosystem memberships overlap, so ecosystem denominators should not be summed. Exact duplicate records across archives were deduplicated, and no divergent same-ID variants were observed. The six selected ecosystems cover major package managers but not every OSV ecosystem. Results should not be generalized to operating-system distributions, containers, firmware, or ecosystems not examined.

7. Conclusions

In this six-ecosystem OSV snapshot, 10.98% of active unique records carried a schema-native quantitative severity item. That pooled value concealed a range from 2.85% in npm to 93.50% in Maven and was dominated by malicious-package records without standard severity. Excluding MAL raised coverage to 75.20%.

The representation details were as informative as the count. Every observed item was a top-level CVSS v3 or v4 vector. Affected-package severity arrays were unused, explicit assessment provenance was absent, and local qualitative labels filled only a small part of the gap. Consumers should keep version, path, provenance, source family, and missingness intact. Any system that converts all records to one severity number without those distinctions will manufacture comparability that the source data do not provide.

Declarations

Ethics approval and consent to participate: Not applicable. The study used public vulnerability metadata and involved no human participants, personal data, exploitation, or scanning of third-party systems.

Consent for publication: Not applicable.

Data and code availability: A blinded reproducibility supplement accompanies the submission. It contains the frozen protocol and amendment, source manifest, analysis code, tests, aggregate outputs, figures, and environment lock. Raw OSV archives are excluded because of size and can be reacquired from the official URLs and verified against the supplied SHA-256 manifest.

Funding: No external funding was received.

Competing interests: The author group is affiliated with the journal's publisher. The group requests handling by an independent editor outside the author group and publisher-management chain. No author-group member will select reviewers or participate in the editorial decision.

Author contributions: Withheld from the blinded manuscript and supplied on the separate title page.

Use of generative AI: Generative AI tools assisted protocol development, code drafting, test design, analysis execution, literature organization, and language drafting. The author group verified the sources, code, outputs, claims, interpretations, and final manuscript and accepts full responsibility for the work.

References

1. OpenSSF Vulnerability Disclosures Working Group. Open Source Vulnerability format, schema v1.9.1. 2026. <https://ossf.github.io/osv-schema/>
2. Google Open Source Security Team. OSV: Open Source Vulnerabilities. <https://osv.dev/>
3. FIRST. Common Vulnerability Scoring System v2.0. <https://www.first.org/cvss/v2/>

4. FIRST. Common Vulnerability Scoring System v3.1: Specification Document. <https://www.first.org/cvss/v3-1/specification-document>
5. FIRST. Common Vulnerability Scoring System v4.0: Specification Document. <https://www.first.org/cvss/v4-0/specification-document>
6. Wunder J, Kurtz A, Eichenmüller C, Gassmann F, Benenson Z. Shedding Light on CVSS Scoring Inconsistencies: A User-Centric Study on Evaluating Widespread Security Vulnerabilities. IEEE Symposium on Security and Privacy. 2024. IEEE document 10646847; arXiv:2308.15259.
7. Croft R, Babar MA, Li L. An Investigation into Inconsistency of Software Vulnerability Severity across Data Sources. SANER. 2022. doi:10.1109/SANER53432.2022.00050.
8. Walkowski M, Nowak M, Balsam A, Nowak K, Oko J, Sujecki S. Fragmentation of CVSS scores in the NVD: A quantitative analysis of inconsistency across vulnerability scoring standards. Computers & Security. 2026;164:104978. doi:10.1016/j.cose.2026.104978.
9. Coutinho LS, Menasche D, Miranda L, Lovat E, Kumar SG, Ramchandran A, Kocheturov A, Limmer T. How Context Impacts Vulnerability Severity: An Analysis of Product-Specific CVSS Scores. Proceedings of LADC. 2024. doi:10.1145/3697090.3697109.
10. Allodi L, Massacci F. Comparing Vulnerability Severity and Exploits Using Case-Control Studies. ACM Transactions on Information and System Security. 2014;17(1). doi:10.1145/2630069.
11. Ruohonen J. A look at the time delays in CVSS vulnerability scoring. Applied Computing and Informatics. 2019;15(2):129–135. doi:10.1016/j.aci.2017.12.002.
12. Decan A, Mens T, Constantinou E. Detection, assessment and mitigation of vulnerabilities in open source dependencies. Empirical Software Engineering. 2021;26. doi:10.1007/s10664-020-09830-x.
13. National Institute of Standards and Technology. National Vulnerability Database. <https://nvd.nist.gov/>
14. Google Open Source Security Team. Announcing a unified vulnerability schema for open source. 24 June 2021. <https://security.googleblog.com/2021/06/announcing-unified-vulnerability-schema.html>