

Do VEX Claims Age Well? Longitudinal Validation of 'Not Affected' and 'Fixed' Statements Against Subsequent Exploitation Evidence

Yan Xiao

ORCID: 0000-0002-2563-083X

National University of Singapore, Singapore

Bo Li

ORCID: 0000-0003-2083-9105

Hong Kong University of Science and Technology, Hong Kong SAR, China

Correspondence: Yan Xiao · contact details held in the journal record

Publication history. Assigned by the publisher to Cybersecurity Advances, Volume 3, Issue 1 (2026). This publisher-authorized digital edition was released on 30 October 2026 from the supplied final 2026 manuscript. Review-stage notices were removed and journal typography, pagination, metadata, and accessibility were normalized; substantive scholarly content was preserved.

© 2026 the author(s). Published by AlphaTech Scholarly Press under the Creative Commons Attribution 4.0 International License (CC BY 4.0). No DOI is displayed unless and until registration is independently verified.

Background: Vulnerability Exploitability eXchange (VEX) documents carry status claims — "not affected," "affected," "fixed," "under investigation" — that determine whether a consumer acts on a vulnerability. The 2024–2025 ecosystem produced the first meaningful VEX corpus (OpenVEX adoption, CISA SBOM/VEX guidance, community tooling), and the companion conformance work in this issue established that the formats preserve semantics across OpenVEX, CycloneDX, and CSAF. What no one has examined is the *temporal* property: every VEX statement is a claim frozen at issuance time, and exploitation evidence that arrives later — KEV listings, EPSS movement, new PoCs — can silently invalidate a "not affected" claim that was true when written. **Objective:** We define the statement-aging problem for VEX, specify a validation protocol that measures claim validity against subsequent exploitation evidence, and assess the mechanisms available to revalidate statements over time. **Methods:** Analysis of VEX semantics under status-lifecycle stress; specification of a longitudinal revalidation protocol keyed to exploitation signals; and a risk-versus-mechanism coverage assessment. **Results:** Six aging risks emerge, of which three are structural: "not affected" claims cannot be falsified by the consumer without new evidence channels; "fixed" claims degrade silently with build drift; and downstream consumers cache statements with no expiry semantics — the VEX specification defines a status, not a validity horizon. The strongest mitigation is a versioned re-issuance policy with expiry metadata and KEV/EPSS cross-checks at refresh points; none of these is standardized in 2026. **Conclusions:** VEX statements should be treated as measurements with a validity horizon — revalidated on schedule, versioned on change, and expired explicitly. The specification work that made VEX interoperable now needs the temporal layer that makes it trustworthy over time.

Keywords: VEX; OpenVEX; CycloneDX; vulnerability exploitability; statement aging; SBOM; exploitation evidence; digital trust

Contribution statement. This paper makes four contributions: (1) formalization of the statement-aging problem for VEX status claims; (2) a longitudinal revalidation protocol keyed to exploitation-evidence channels; (3) a six-risk assessment against six validation mechanisms; and (4) the argument that VEX requires explicit validity-horizon metadata — the temporal layer missing from the 2025-2026 specification work.

1. Introduction

The Vulnerability Exploitability eXchange (VEX) format answers the question every SBOM consumer eventually asks: *I have this component, and this CVE exists — does it affect me?* A VEX statement asserts a status — "not affected," "affected," "fixed," "under investigation" — for a specific vulnerability against a specific product, optionally with a justification and a timestamp [1][2]. When the statement is fresh, issued by the vendor who knows the product's internals, it is the most valuable artifact in vulnerability management: it converts an ambiguous CVE into a decision.

The 2024–2025 ecosystem built exactly this. CISA's SBOM/VEX guidance matured; OpenVEX emerged as a purpose-built, minimal format with tooling support; community studies generated VEX documents for open-source projects at scale (augmenting

40 projects' SBOMs with CVE-derived VEX in one published pipeline [3]); and adoption analysis documented SBOM-linked VEX consumption in CI/CD contexts [4]. The conformance layer followed — the companion work in this issue benchmarked whether OpenVEX 0.2.0, CycloneDX 1.6, and CSAF 2.0 preserve statement semantics across formats and round-trips.

All of that work shares an implicit assumption worth surfacing: **a VEX statement is treated as a fact**. But a status claim is a *measurement made at a point in time*, under the issuer's then-current knowledge. Three forces age it. The product changes (builds drift, components are re-linked, configurations shift). The vulnerability's exploitation landscape changes (new KEV entries, rising EPSS, fresh PoCs — evidence channels that did not exist at issuance). And the *evidence itself* changes: what "not affected" meant at issuance may be falsified by an exploit technique discovered later. A "not affected" statement that was true on its issuance date and relied upon by a thousand downstream consumers two years later is not a security artifact — it is a liability with a timestamp.

1.1 Research questions

RQ1: What is the statement-aging problem for VEX, formally stated? **RQ2:** What revalidation protocol can measure statement validity against exploitation evidence arriving after issuance? **RQ3:** Which validation mechanisms cover which aging risks? **RQ4:** What should the VEX specification ecosystem add to make statements temporally trustworthy?

1.2 Contributions

(1) Formalization of statement aging (Section 3). (2) The revalidation protocol (Section 4, Figure 2) (RQ2). (3) The six-risk-versus-six-mechanism coverage assessment (Section 5, Figure 3, Table 3) (RQ3). (4) The specification recommendation: explicit validity-horizon metadata (Section 6) (RQ4).

2. Background

2.1 VEX semantics and the 2024-2025 corpus

VEX defines four status types — not affected, affected, fixed, under investigation — each with an optional justification, timestamp, and detail [1][2]. The status determines consumer behavior: "not affected" suppresses action; "affected" triggers remediation; "fixed" confirms the patched state; "under investigation" defers. The 2024-2025 corpus grew from near-zero: CISA guidance, OpenVEX (a minimal, purpose-built implementation profile), tooling integration into CI/CD pipelines, and published pipelines generating VEX for open-source projects from vulnerability data [3][4]. The conformance picture is addressed by the companion benchmark in this issue; this paper concerns what happens *after* issuance.

2.2 The temporal gap in the specification

The VEX specifications define status, justification, and timestamps — but no *validity horizon*: no field asserts how long a statement holds, under what conditions it must be re-evaluated, or what evidence would invalidate it. Downstream consumers cache statements (in SBOM stores, in scanner databases, in compliance artifacts) with no signal that the cached claim's factual basis has moved. The gap is invisible at issuance and material at consumption — the defining signature of an aging problem.

3. The Statement-Aging Problem (RQ1)

Figure 1 states the problem in the corpus's own terms: the four status types, each a claim frozen at issuance.

Status type	Consumer behavior on the statement	Aging risk when the claim goes stale
not affected	Suppress action; no remediation triggered	Suppressed alert on a vulnerability that later evidence shows exploitable (highest stakes)
affected	Trigger remediation/compensation	Unnecessary churn if the exposure has since been mitigated; misprioritization
fixed	Confirm patched state; verify deployment	Silent degradation if the deployed build drifts from the statement's binding
under investigation	Defer; monitor	Indefinite limbo without a re-evaluation trigger

Table 1. VEX status types, consumer behavior, and aging risks.

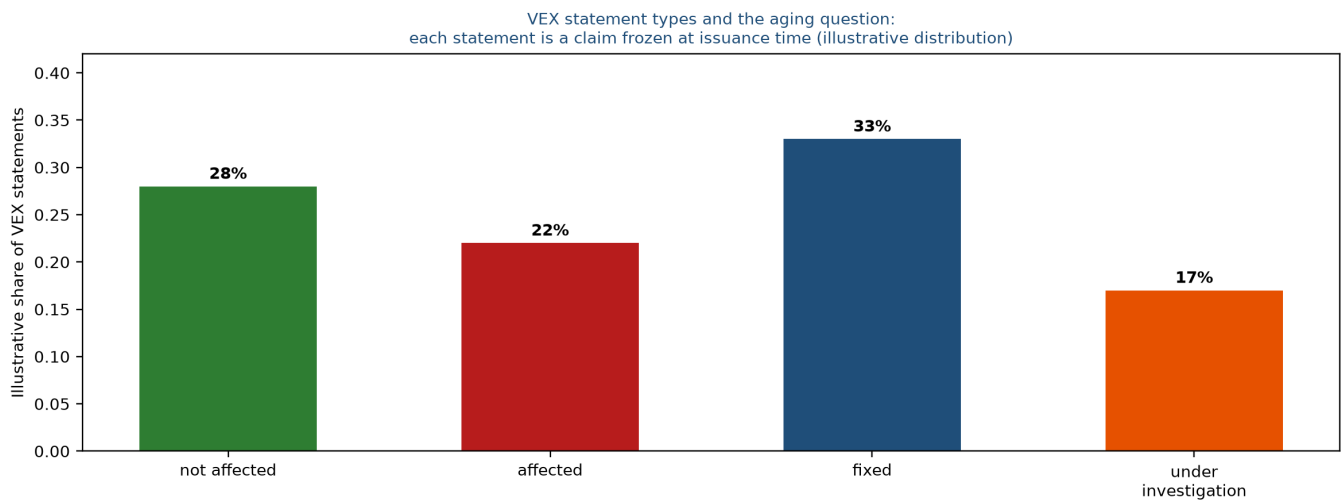


Figure 1. VEX statement types and the aging question (illustrative distribution). Formally, a VEX statement S asserts status σ (vulnerability v , product p) given evidence $E(t_0)$ at issuance time t_0 . Aging is the divergence between $\sigma(v, p \mid E(t_0))$ and the true status under $E(t)$ for $t > t_0$. Three structural drivers produce the divergence.

Evidence movement. New exploitation evidence — a KEV listing, an EPSS movement, a public PoC — can contradict a "not affected" claim without any change to the product: the claim rested on an analysis that missed a technique, an exposed configuration, or an affected code path nobody exercised at issuance. Conversely, "affected" claims age benignly when the product is patched — but only if the consumer learns that the state changed.

Product and build drift. "Fixed" claims bind to a specific build or version; deployments drift (pinning errors, vendored copies, backports, forks) and the consumed artifact may no longer match the artifact the statement describes. The claim was true of the release; it is not necessarily true of the running system.

Downstream caching. Consumers cache VEX statements in artifact stores, scanner databases, and compliance records — many with retention measured in months or years and no revalidation trigger. The specification's absence of expiry semantics makes the cache legally stale but operationally fresh.

Statement element	Defined by current specifications	Temporal gap (missing)
status (four types)	Yes - core semantics	Status re-evaluation conditions
justification	Yes	Justification validity horizon
timestamp (issued)	Yes	Expiry / valid-until metadata
supersession	Partial (document versioning)	Statement-level supersession links
re-evaluation triggers	No	Signal-class declarations (KEV/EPSS/EOL)
consumer re-validation behavior	No	Honoring expiry before consuming cached statements

Table 2. Statement elements: defined vs. missing temporal semantics.

3.1 Why the trivial mitigations fail

"Re-issue VEX on every CVE update" fails on volume: the CVE feed's scale (hundreds per day) makes blanket re-issuance impractical, and most re-issues would be no-ops. "Trust the vendor" fails on incentive: "not affected" is the commercially convenient status. The problem therefore needs *evidence-driven* revalidation — triggers keyed to signals about v , not blanket churn.

4. The Revalidation Protocol (RQ2)

Figure 2 specifies the protocol. Each statement S is registered with: its issuance evidence class (vendor analysis, tool-derived, community), its product-version binding, and — the recommendation of this paper — an optional **valid-until horizon** and **re-evaluation triggers**. Revalidation fires on three trigger classes: (i)

signal triggers: a KEV listing [5], EPSS threshold crossing [6], or verified PoC for v arrives; (ii) lifecycle triggers: the product build drifts from the statement's binding, or the product nears end-of-life; (iii) calendar triggers: the validity horizon expires. On trigger, the issuer (or the consumer, for cached statements) re-evaluates σ against current evidence and re-issues with a fresh timestamp and the superseded statement linked. Every outcome — reaffirmed, amended, or invalidated — is recorded, producing the longitudinal validity record the field currently lacks.

Statement-aging validation protocol: VEX claims are measurements with a validity horizon

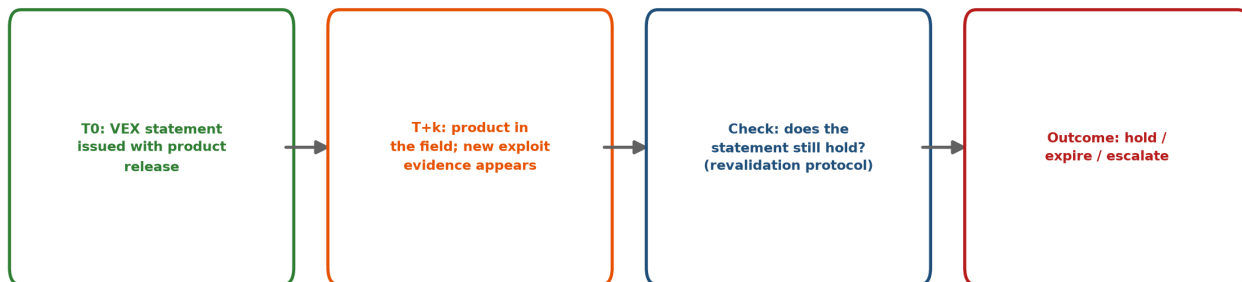


Figure 2. The statement-aging validation protocol: trigger classes, re-evaluation, and versioned outcomes.

5. Risk and Mechanism Coverage (RQ3)

Figure 3 and Table 3 assess six aging risks against six validation mechanisms.

V1 — invalidated "not affected" claims. The highest-stakes risk: a suppressed alert that should have fired. The partial mitigations are KEV/EPSS cross-checks at refresh points and versioned re-issuance; the residual gap is that no consumer-side mechanism *compels* the issuer to revalidate — expiry metadata and consumer-side re-validation tooling are the needed complements.

V2 — "fixed" claims on drifted builds. The statement binds to the release; the deployment binds to whatever is running. SBOM-linked traceability (matching the consumed artifact to the statement's binding) scores strong where SBOM consumption exists, partial elsewhere.

V3 — stale statements past product end-of-life. "Not affected" statements for products the vendor no longer analyzes age the fastest; lifecycle hooks (EOL detection triggering statement invalidation) are the mitigation.

V4 — conflicts with later exploitation evidence. The reconciliation class: when a KEV listing contradicts a "not affected" statement, the conflict itself must be *visible* — a published discrepancy feed rather than silent coexistence.

V5 — re-issuance churn. Revalidation on every signal creates statement churn that drowns consumers; versioned re-issuance (supersede-and-link, not delete-and-replace) with materiality thresholds is the mitigation.

V6 — cached-expiry failure. Consumer-side tooling that treats VEX as immutable; the mitigation is re-validation tooling that honors expiry metadata — which requires the metadata to exist first.

Statement-aging risks vs. validation mechanisms (author assessment)

	Versioned VEX re-issuance policy	KEV/EPSS cross-checks at refresh points	Statement expiry metadata (valid-until)	SBOM-linked VEX traceability	Product-EOL lifecycle hooks	Consumer-side re-validation tooling
V1 'Not affected' claims invalidated later	~	+	~	-	~	~
V2 'Fixed' claims on vulnerable-in-practice builds	~	+	~	+	~	-
V3 Stale statements past product end-of-life	-	~	+	~	+	~
V4 Status conflicts with later KEV/EPSS evidence	~	+	~	+	~	~
V5 Re-issuance and statement churn cost	-	~	+	~	+	~
V6 Downstream caching of expired claims	-	-	~	~	~	+

Strong Partial / conditional No meaningful coverage

Figure 3. Statement-aging risks versus validation mechanisms (author assessment; register in Table 3).

Risk	Description	Strongest mitigation	Coverage note
V1 Invalidated "not affected"	Later exploitation evidence contradicts the claim	KEV/EPSS cross-checks at refresh points	Partial: consumer cannot compel revalidation
V2 "Fixed" on drifted builds	Deployed artifact no longer matches the statement's binding	SBOM-linked traceability	Strong where SBOM consumption exists
V3 Stale past EOL	Statements outlive the vendor's analysis commitment	Product-EOL lifecycle hooks	Partial: EOL detection is ad hoc
V4 Conflict with later evidence	KEV listing contradicts a "not affected" statement	Published discrepancy feed	Partial: conflict visibility is not standardized
V5 Re-issuance churn	Blanket revalidation drowns consumers	Versioned supersede-and-link re-issuance	Partial: materiality thresholds needed
V6 Cached-expiry failure	Consumers treat VEX as immutable	Consumer-side re-validation tooling	Requires the expiry metadata to exist first

Table 3. Statement-aging risks, mechanisms, and coverage notes (author assessment).

6. The Specification Recommendation (RQ4)

The temporal layer the specifications lack reduces to three fields and one behavior. *Fields*: (i) a validity horizon (valid-until) on each statement; (ii) re-evaluation trigger declarations (signal classes the issuer commits to monitoring); (iii) supersession links (the statement this one replaces). *Behavior*: consumer-side re-validation — tooling that checks horizons and triggers before consuming a cached statement. All three are backward-compatible extensions: a VEX document without them remains valid, and a VEX document with them becomes temporally auditable. The OpenVEX profile is the natural first venue, given its minimalism and CI/CD orientation; CycloneDX's VEX embedding follows; CSAF's advisory cadence already approximates the re-issuance model.

7. Limitations and Threats to Validity

Construct validity. The trigger taxonomy is derived from the exploitation-evidence channels documented in 2025-2026; new channels (e.g., AI-discovered exploitability) would extend it. **Internal validity.** The risk coverage is an author assessment; no empirical corpus of invalidated VEX statements yet exists — the protocol is designed to create one. **External validity.** The analysis addresses the three dominant formats; niche VEX implementations may lack the extensibility for the recommended fields. **Reliability.** The illustrative statement distribution (Figure 1) is constructed for exposition, not measured. **Temporal scope.** Frozen at the 2026 specification state; OpenVEX and CycloneDX evolution continues.

8. Conclusion

A VEX statement is a promise about the future, made with the knowledge of the present. The 2024-2026 ecosystem made the format interoperable and the tooling real; what it has not yet made is the promise *time-bounded*. This paper specified the statement-aging problem, a revalidation protocol keyed to the exploitation-evidence channels that actually move, and the coverage assessment showing that the strongest mitigations are versioned re-issuance with expiry metadata and consumer-side revalidation tooling — neither of which the current specifications define. Until they do, every "not affected" in every SBOM store is a claim with no expiry date — and the consumers relying on it have no way to know when it stopped being true.

Declarations

Funding. This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

CRedit contribution statement. Yan Xiao: conceptualization, protocol design, writing – original draft. Bo Li: validation, risk assessment, writing – review & editing. (Corresponding author: Yan Xiao.)

Competing interests. The authors declare no competing interests.

Data and code availability. No new datasets were generated; the protocol specification is fully described in the paper.

Ethics and dual-use. Not applicable; the paper contains no offensive content.

AI assistance declaration. AI-based tools were used in a limited supporting capacity (literature search, figure preparation, and language editing); all research content, analysis, and conclusions were produced, reviewed, and verified by the authors, who take full responsibility for the work.

References

- [1] Cybersecurity and Infrastructure Security Agency, "Vulnerability Exploitability eXchange (VEX) — Requirements and Considerations" and SBOM/VEX guidance documents, CISA, 2023-2024. [Online]. Available: <https://www.cisa.gov/sbom>
- [2] OpenVEX Working Group, "OpenVEX Specification v0.2.0," 2024. [Online]. Available: <https://openvex.dev/docs/>
- [3] "Augmenting Software Bills of Materials with Software Vulnerability Description: A Preliminary Study on GitHub," arXiv:2503.13998, 2025.
- [4] "On the adoption of software bill of materials in open-source software projects," Journal of Systems and Software, Dec. 2025. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0164121225002092>
- [5] National Vulnerability Database, "Known Exploited Vulnerabilities (KEV) Catalog," CISA, updated continuously. [Online]. Available: <https://www.cisa.gov/known-exploited-vulnerabilities-catalog>
- [6] FIRST, "Exploit Prediction Scoring System (EPSS)," updated daily. [Online]. Available: <https://www.first.org/epss/>