A CycloneDX VEX Profile for FedRAMP Rev5 VDR/VER Disposition and Response

Matthew Venne
Chief Technology Officer, stackArmor

June 2026

Abstract

This is a companion to *A Deterministic, CVSS–Environmental Method for FedRAMP Rev5 VDR/VER Vulnerability Prioritization*, which assigns every detected finding a Potential Agency Impact rating (PAIN, N1–N5) and a remediation deadline by formula. That scoring is deterministic and computable *a priori*. Deciding whether a finding is *truly* exploitable in the running environment — and, when a fix will miss the deadline, attesting the compensating response — is a separate, *a posteriori* step that can only follow detection. FedRAMP's Vulnerability Evaluation Requirements (VER) make that step normative but name no format. This proposal recommends **CycloneDX VEX** as the structured, machine-readable, signable carrier for it, chosen for the fit of its disposition and response *vocabulary* — not because it carries scores; it does not. We give the field mapping to VER, a thin worked example, an OCI-native distribution pattern, a deployment-context service/workload attestation pattern, and an optional crosswalk to CISA's status-justification labels for downstream interoperability. This document is an *implementation profile*: it shows one defensible way to satisfy existing VER requirements; it does not propose changing them. The key words MUST, SHOULD, and MAY are to be read as in RFC 2119.

1 Where disposition sits in the lifecycle

The method memo computes PAIN and a deadline for *every* finding a scanner emits. But a scan hit is a candidate, not a confirmed exposure: the flagged code may be absent, may never execute, may require a configuration nobody enabled, or may sit behind a control that already neutralizes it. Resolving which is the *disposition* step.

Disposition is an *a posteriori* determination. The factors that decide it — reachability, configuration, surrounding mitigations — are specific to a given vulnerability on a given deployment and cannot be enumerated before that vulnerability is known. The analysis therefore necessarily follows detection, acutely so for novel vulnerabilities, where no prior due-diligence could exist. This is a statement about *timing*, not about who or what performs it: the determination MAY be fully automated or AI/agentic, but it cannot be precomputed.

Because VER's clocks are short, the determination must be captured in a form that machines can produce, exchange, sign, and act on. A disposition resolves a finding onto one of two tracks:

- **Not truly affected** — the finding is suppressed, with a machine-readable justification recording *why* (code absent, unreachable, configuration-gated, or already mitigated).
- **Affected but remediation will be late** — the finding is accepted within process, with an attested mitigation and a planned response, rather than silently breaching the deadline.

2 Why CycloneDX: the vocabulary fits the lifecycle

FedRAMP is not bound to any single VEX format, and VER specifies *fields and content*, not a schema (Section 3). The provider therefore chooses the carrier — the same latitude FedRAMP itself exercised in substituting PAIN for SSVC. On that basis the deciding factor is *expressiveness*: which format's disposition vocabulary most precisely captures the real outcomes above. CycloneDX's `analysis` object is the strongest fit. The same granularity pays off when disposition is automated: because each justification names a distinct precondition, an automated or AI/agentic analyzer can map each evidence source one-to-one to a justification, where a coarser vocabulary would collapse several distinct findings onto a single label.

CycloneDX field	**Values (and what they capture)**
`analysis.state`	`resolved`, `resolved_with_pedigree`, `exploitable`, `in_triage`, `false_positive`, `not_affected` — note a first-class `false_positive` and an `in_triage` (work-in-progress) state.
`analysis.justification`	`code_not_present`, `code_not_reachable`, `requires_configuration`, `requires_dependency`, `requires_environment`, `protected_by_compiler`, `protected_at_runtime`, `protected_at_perimeter`, `protected_by_mitigating_control` — nine reasons, naming configuration, reachability, and perimeter/runtime controls explicitly.
`analysis.response`	`can_not_fix`, `will_not_fix`, `update`, `rollback`, `workaround_available` — the planned action, which the late-remediation track needs and lighter formats lack.
`analysis.detail`	free text for human-readable context and evidence.

Table 1: CycloneDX's `analysis` block: states, justifications, and responses.

The operational cases a CSP actually encounters map onto native values without contortion:

Real-world disposition	**CycloneDX expression**
Scanner false positive	`state: false_positive`
Vulnerable code absent / removed	`not_affected` + `code_not_present`
Vulnerable path never executes	`not_affected` + `code_not_reachable`
Needs an unset configuration	`not_affected` + `requires_configuration`
Neutralized at the edge (WAF/IAP)	`not_affected` + `protected_at_perimeter`
Neutralized at runtime / by a control	`not_affected` + `protected_at_runtime` / `protected_by_mitigating_control`
Affected; fix will miss deadline	`exploitable` + `response:[workaround_available, update]` + `detail`
Under evaluation	`in_triage`

Table 2: Operational dispositions and their native CycloneDX encodings.

3 Mapping to FedRAMP VER

VER requires the disposition step and a machine-readable report, in each case by function rather than by named format. CycloneDX satisfies each clause:

VER requirement	CycloneDX mechanism
`VER-EVA-EFP` (evaluate false positives)	`analysis.state = false_positive`
`VER-EVA-AIA` (assume exploitable absent evidence)	a VEX statement *is* the "evidence otherwise"; its absence leaves the assume-exploitable default in force
`VER-RPT-VDT` (per-finding disposition)	`analysis.state` + `justification` + `response` keyed to the CVE and component
`VER-TFR-MAV` / `VER-RPT-AVI` (accepted vulns)	`state = exploitable` with `response` and `detail` recording the acceptance and mitigation
`VER-TFR-MRH` (machine-readable JSON history)	CycloneDX is JSON; statements accrete over time with timestamps

Table 3: FedRAMP VER clauses and the CycloneDX fields that satisfy them.

4 A thin attestation: what it carries, and what it does not

The disposition artifact stays deliberately small. It carries only what identifies the finding and records the judgment:

- the vulnerability `id` and its `source`;
- the affected subject, by `affects[].ref`: either a software component `bom-ref` (for artifact applicability) or a service/workload `bom-ref` (for deployment-context reachability);
- the `analysis` block — `state`, `justification`, `response`, `detail`;
- provenance — author, timestamp, and a signature over the artifact.

It deliberately *omits* PAIN, the LEV/IRV axes, the Certification Class, and the CVSS vector. Those are computed deterministically by the method and live in the provider's scoring report; duplicating them into the disposition artifact would invite drift and tempt a reader to treat an attestation as a place to re-score. The two records are joined on a stable key — the **CVE identifier plus the affected subject reference** — so a reporting layer can attach each disposition to the finding it governs and recover the retained PAIN for audit. The disposition changes a finding's *handling*; it never changes its score.

5 Service and workload subjects: reusable reachability inputs

CycloneDX can describe both software `components` and network-facing `services`. Its vulnerability `affects[].ref` field references a subject by `bom-ref`, and that subject can be a service as well as a package or image component. This matters for FedRAMP reachability because the companion reachability model keeps three facts separate:

- a **direct-surface attestation** says which network descriptors the internet can deliver to a deployed service or workload — the set $N(a)$;
- a **content-receipt assertion** records whether the component receives internet-derived content — the coarse predicate $C_0(a)$; and
- a **finding disposition** says whether a particular vulnerability's required direct surface matches $N(a)$ or a confirmed indirect trigger applies to $C_0(a)$ and remains after the current prevention assertion $P_0(v, a)$ is applied.

A service-level input attestation is therefore reusable across findings. It does not, by itself, declare every CVE on the service reachable or unreachable; instead it caches the asset-side evidence

so each CVE can be evaluated automatically without rediscovering Ingress, Gateway, load-balancer, backend protocol, ALPN, and edge-auth state. If a service attestation shows no direct internet surface, it closes only the direct branch; a confirmed indirect content trigger can still make a finding IRV. If a direct surface exists, `AV:N` remains an enrichment hint rather than the decision: the finding's required protocol and other descriptors still need to match, with unresolved required surface treated conservatively for that direct branch.

A stable service/workload `bom-ref` SHOULD identify the deployment subject, not merely the image artifact. A Kubernetes service, for example, can be represented with a stable URN-like reference:

```
{
  "services": [
    {
      "bom-ref": "urn:k8s:prod:default:service:payments-api",
      "name": "payments-api",
      "endpoints": ["https://api.example.com"],
      "properties": [
        { "name": "k8s:cluster", "value": "prod" },
        { "name": "k8s:namespace", "value": "default" },
        { "name": "k8s:kind", "value": "Service" },
        { "name": "k8s:name", "value": "payments-api" },
        { "name": "vdr:directInternetSurfacePresent", "value": "true" },
        { "name": "vdr:receivesInternetDerivedContent", "value": "true" },
        { "name": "vdr:exposedBackendProtocol", "value": "http" },
        { "name": "vdr:exposedBackendProtocolVersion", "value": "HTTP2" },
        { "name": "vdr:exposedBackendAlpn", "value": "h2" },
        { "name": "vdr:routeEvidence", "value": "HTTPRoute public/payments -> Service
     default/payments-api:8080" }
      ]
    }
  ]
}
```

The per-finding record can then reference the same service subject and record the automated disposition for a specific CVE:

```
{
  "id": "CVE-2026-30303",
  "analysis": {
    "state": "exploitable",
    "detail": "A direct internet surface exists and CVSS is AV:N. Required protocol is unknown, so the
     direct required surface is treated conservatively."
  },
  "affects": [{ "ref": "urn:k8s:prod:default:service:payments-api" }],
  "properties": [
    { "name": "vdr:findingInternetReachable", "value": "true" },
    { "name": "vdr:reachabilityDecision", "value": "direct_surface_match" },
    { "name": "vdr:remediationTrack", "value": "LEV+IRV" }
  ]
}
```

This split preserves automation without overclaiming. The service attestation is reused for all CVEs on that deployed subject, while the per-CVE disposition still accounts for `AV`, required protocol, ALPN/version, confirmed indirect-trigger status, C_0, P_0, and other vulnerability-specific preconditions. The attestation SHOULD be re-evaluated when the facts that justify it change: Service selector, selected port, Ingress or Gateway binding, backend protocol, ALPN policy, edge authentication, or the workload template/image digest when that digest is part of the evidence. Routine replica churn need not invalidate the attestation when those evidence inputs remain unchanged. Content-receipt assertions should also change when application handoffs change; P_0 evidence is

maintained and invalidated under the companion reachability model rather than inferred from the direct-surface record.

6 Worked example

A single signed CycloneDX document can carry several statements. Below, one finding is dispositioned `not_affected` (unreachable code) and one is `exploitable` with an attested perimeter mitigation because the vendor fix will arrive after the deadline. Note the absence of any score, rating, or PAIN field.

```
{
  "bomFormat": "CycloneDX",
  "specVersion": "1.6",
  "version": 1,
  "metadata": {
    "timestamp": "2026-06-28T14:00:00Z",
    "authors": [{ "name": "CSP Security Team" }],
    "component": {
      "type": "application",
      "name": "payments-api",
      "bom-ref": "payments-api@2.4.1"
    }
  },
  "vulnerabilities": [
    {
      "id": "CVE-2026-10101",
      "source": { "name": "NVD",
        "url": "https://nvd.nist.gov/vuln/detail/CVE-2026-10101" },
      "analysis": {
        "state": "not_affected",
        "justification": "code_not_reachable",
        "detail": "Vulnerable parser entrypoint is never invoked; only the
                   signing helper from this library is linked."
      },
      "affects": [{ "ref": "pkg:golang/example.com/parser@1.2.3" }]
    },
    {
      "id": "CVE-2026-20202",
      "source": { "name": "NVD",
        "url": "https://nvd.nist.gov/vuln/detail/CVE-2026-20202" },
      "analysis": {
        "state": "exploitable",
        "response": ["workaround_available", "update"],
        "detail": "Vendor fix ETA exceeds the VER deadline. Perimeter WAF
                   virtual patch deployed and egress restricted; tracked for
                   update on vendor release."
      },
      "affects": [{ "ref": "pkg:deb/debian/libxyz@2.1.0-1" }]
    }
  ]
}
```

7 Distribution and retrieval over OCI

For artifact-only dispositions, the VEX document SHOULD travel with the image it describes. CycloneDX documents have a defined media type and can be stored as OCI artifacts in the same registry as the image and associated with it through the registry's referrers/attestation mechanism. A scanner that already pulls an image to detect vulnerabilities can, in the same pass, resolve the image's attached VEX and apply the dispositions automatically.

Deployment-context reachability attestations have a different subject: the service, workload, or runtime component. They MAY still be distributed as signed OCI artifacts, but the reference is to a stable service/workload `bom-ref` and the evidence binds the statement to the route and backend protocol configuration that justified it. This allows the same direct-surface and content-receipt inputs to be reused across scans and CVEs while still forcing re-evaluation when the deployment surface changes. The result is the same operational property as image-attached VEX: the *a posteriori* judgment, once made and signed, is rediscovered and reused without re-litigation, but without pretending that deployment-context reachability is a property of the image alone.

8 Optional interoperability: CycloneDX to CISA labels

A FedRAMP-facing report needs nothing beyond the above. Some downstream consumers — agency tooling, or a CSAF advisory pipeline — speak CISA's narrower status-justification vocabulary instead. CISA, like CycloneDX, separates the two axes: each CycloneDX `state` coarsens onto a CISA *status* and each `justification` onto a CISA *justification label*, in both cases without loss of safety (the mapping only ever generalizes; it never invents a stronger claim). This crosswalk is OPTIONAL and exists solely for interchange.

CycloneDX justification / state	**CISA label (coarsened)**
`code_not_present`	`vulnerable_code_not_present`
`code_not_reachable`	`vulnerable_code_not_in_execute_path`
`requires_configuration`	`vulnerable_code_not_in_execute_path`
`requires_dependency`	`vulnerable_code_not_in_execute_path`
`requires_environment`	`vulnerable_code_not_in_execute_path`
`protected_by_compiler`	`inline_mitigations_already_exist`
`protected_at_runtime`	`inline_mitigations_already_exist`
`protected_at_perimeter`	`inline_mitigations_already_exist`
`protected_by_mitigating_control`	`inline_mitigations_already_exist`
`state: not_affected`	`not_affected` (paired with the justification above as the CISA label)
`state: false_positive`	`not_affected` (`component_not_present` where the component is truly absent)
`state: in_triage`	`under_investigation`
`state: exploitable`	`affected`
`state: resolved` / `resolved_with_pedigree`	`fixed`

Table 4: Optional coarsening of CycloneDX dispositions onto CISA's labels/statuses.

9 Standing and adoption

CycloneDX introduced its vulnerability and VEX capabilities in version 1.4 (January 2022), ahead of CISA's status-justification guidance (June 2022); its richer vocabulary is thus an independent, earlier design rather than a divergence from a later one. CycloneDX is now an Ecma International standard (ECMA-424). Adoption is additive: the disposition layer sits on top of the unchanged scoring pipeline of the method memo, joined by CVE and component, and a provider MAY emit the optional CISA crosswalk for consumers that require it. As with the method's archetype catalog, this profile is offered as a workable existence proof, not as a new FedRAMP standard — the choice of carrier remains the provider's.