GCVE BCP-05-X-04 - Security Vulnerability and Advisory Timeline

GCVE.eu

GCVE.eu

Contents

1 GCVE BCP-05-X-04: Security Vulnerability and Advisory Timeline **1**

1.1 Abstract . . . 1
1.2 Motivation and Scope . . . 2
1.3 Placement . . . 2
1.4 Data Model . . . 3
1.4.1 Metadata . . . 3
1.4.2 Event Fields . . . 4
1.4.3 Event Kinds . . . 7
1.4.4 Dates, Ordering, and Duplicates . . . 8
1.5 Metric Evidence . . . 8
1.5.1 Roles and Evidence Bases . . . 8
1.5.2 Eligibility and Source Mapping . . . 9
1.6 Duration Metrics . . . 11
1.6.1 Anchor Selection . . . 12
1.6.2 Defined Metrics . . . 13
1.6.3 Metric Fields . . . 14
1.6.4 Precision and Bounds . . . 16
1.7 Example Timeline . . . 16
1.8 Processing and Validation Requirements . . . 18
1.9 Updates and Interoperability . . . 19
1.10 Privacy, Security, and Trust Considerations . . . 20
1.11 Example JSON Paths . . . 20
1.12 Reference Implementation . . . 20

1 GCVE BCP-05-X-04: Security Vulnerability and Advisory Timeline



GCVE.eu

- **Version**: 1.0
- **Status**: Draft
- **Date**: 2026-11-17
- **Authors**: GCVE Working Group
- **BCP Extended ID**: BCP-05
- **BCP ID**: BCP-05-X-04

This guide is distributed and available under CC-BY-4.0.

Copyright (C) 2026 GCVE Initiative.

1.1 Abstract

This document defines an extension to GCVE BCP-05 for exchanging a chronological view of security vulnerability and advisory information collected from multiple sources. It preserves dated fields from vulnerability records and linked advisories, sightings, and known exploited vulnerability (KEV) assertions, together with their provenance and original date values.

The extension also defines optional duration metrics whose selected evidence, calculation time, precision, and limitations remain available to consumers. The payload follows the `metadata`, `data`, and `metrics` structure exported by the Vulnerability Lookup implementation.

1.2 Motivation and Scope

A vulnerability's history can be spread across a CVE record, vendor advisories, CSAF documents, other vulnerability databases, sightings, and KEV catalogs. A combined timeline helps consumers inspect publication history, compare sources, and assess the intervals between discovery, reporting, disclosure, exploitation evidence, and patch evidence.

A date found in a linked record is not necessarily a fact about the requested vulnerability. An advisory can cover multiple vulnerabilities, acquire a new vulnerability in a later revision, or describe another issue altogether. A catalog addition date can reflect a backlog import rather than the beginning of exploitation. This extension therefore separates event collection from eligibility for metric calculations.

BCP-05-X-03 represents typed handling, coordination, disclosure, and regulatory events supplied by a publisher. X-04 represents an aggregated view of dated source information and optional derived measurements. X-03 events can be inputs to X-04. The extensions MAY coexist and MUST NOT be treated as interchangeable payloads. X-04 does not define current handling workflow state or establish regulatory compliance.

The key words MUST, MUST NOT, REQUIRED, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL in this document are to be interpreted as described in RFC 2119 and RFC 8174 when they appear in capitals.

1.3 Placement

The extension MUST be placed in the `extensions` object of a GCVE BCP-05 record using the key `bcp-05-x-04`. Its value MUST contain one `x_timeline` object:

```
{
  "x_gcve": [
    {
      "vulnId": "GCVE-1-2026-12345",
      "recordType": "advisory",
      "extensions": {
        "bcp-05-x-04": {
          "x_timeline": {
            "metadata": {
              "vulnerability": "GCVE-1-2026-12345",
              "count": 0,
              "kinds": {"milestone": 0, "sighting": 0, "kev": 0, "other
                  ": 0},
              "linked_omitted": 0
            },
```

```
            "data": []
          }
        }
      }
    }
  ]
}
```

`x_timeline` is scoped by its extension key. In X-04 it contains `metadata`, `data`, and optionally `metrics`; in X-03 it contains handling `events` and optionally `entities` and `regulatory` information.

A standalone timeline export MAY contain the `x_timeline` value directly, without the GCVE wrapper. To embed a Vulnerability Lookup timeline export, a producer places that export under `extensions.bcp-05-x-04.x_timeline`. This does not replace the surrounding BCP-05 record or its required members.

1.4 Data Model

Member	Type	Required	Description
`metadata`	object	yes	Timeline subject, event counts, and collection limitations.
`data`	array of objects	yes	Collected dated events, ordered oldest first. An empty array is allowed.
`metrics`	object	no	Selected evidence anchors and calculated durations.

1.4.1 Metadata

Member	Type	Required	Description
`vulnerability`	string	yes	Identifier of the vulnerability or advisory whose timeline was requested.
`count`	integer	yes	Number of events in `data`; MUST be non-negative.
`kinds`	object	yes	Counts by event `kind`. Values MUST be non-negative integers.
`linked_omitted`	integer	no	Number of linked records excluded from collection, for example because of a resource budget.

`metadata.vulnerability` MUST identify the containing record's subject or a known identifier for the same subject. It MAY be a CVE identifier when the containing GCVE record describes that CVE. An arbitrary relationship to another record does not establish identifier equivalence.

`count` MUST equal the length of `data`. Producers MUST include the four defined kind counts, including zero values; any additional kinds present in `data` MUST also be counted. The sum of all kind counts MUST equal `count`.

`linked_omitted`, when present, MUST be non-negative. A positive value indicates that the collection excludes linked records and that metrics may change when those records are included. Zero indicates no known omission from that collection step; it does not establish complete knowledge of the vulnerability. Absence means the omission count was not supplied. Other collection limits SHOULD be documented by the producer.

1.4.2 Event Fields

Each member of `data` describes a date obtained from a source. Its presence states that the source contains that date or assertion; it does not independently verify the underlying claim.

Member	Type	Required	Description
`timestamp`	string	yes	Normalized RFC 3339 UTC date-time, using `Z`.
`date_only`	boolean	yes	Whether the original value supplied a calendar date without a time.
`value`	string	yes	Original source date value, before normalization.
`kind`	string	yes	Broad event grouping: `milestone`, `sighting`, `kev`, or `other`.
`label`	string	yes	Human-readable event description, such as `Published` or `Vendor fix`.
`source`	string	yes	Source feed, publisher, catalog, or sighting origin. It need not be a URL.
`record`	string	yes	Identifier of the source record, which can differ from the timeline subject.
`field`	string	yes	Path to the date field within the source record.
`linked_record`	boolean	no	**`true`** when the date was collected from a linked record.

Member	Type	Required	Description
`uuid`	string	no	Identifier of the underlying sighting or KEV entry.
`content`	string	no	Source text providing context.
`text`	string	no	Source timeline entry text, for example a CVE free-text timeline description.
`url`	string	no	HTTP or HTTPS reference to the source or supporting information.
`status`	object	no	KEV assertion status, including `exploited` and `status_reason` when available.
`metric_role`	string	no	Structured role for which this event is eligible as calculation evidence.
`metric_basis`	string	conditional	Evidence basis; REQUIRED when `metric_role` is present.
`batch_listing`	boolean	no	Whether the date appears to represent a catalog batch listing or backlog import.

Member	Type	Required	Description
`publication_copy`	boolean	no	Whether an apparent KEV date appears to copy a vulnerability publication timestamp.
`added_in_revision`	boolean	no	Whether the date identifies a revision that added the subject vulnerability to an advisory.

`field` uses dot-separated object member names and zero-based array indices, for example `vulnerabilities.0.remediations.0.date`. It is a descriptive source path, not a JSON Pointer or executable expression. Member names containing dots can be ambiguous; consumers MUST NOT execute or blindly evaluate this string.

`source`, `record`, `field`, and, where supplied, `uuid` SHOULD allow an analyst to trace an event to its input. `source` identifies provenance, not a trust level. `linked_record: true` alone does not make an event eligible for metrics.

1.4.3 Event Kinds

Kind	Meaning
`milestone`	Recognized record milestones, such as identifier reservation or publication.
`sighting`	A dated sighting associated with the subject or an equivalent identifier.
`kev`	A dated field from a KEV assertion or its evidence.
`other`	Another collected date, including discovery, remediation, revision, or document generation dates.

Kinds support grouping and display. A `milestone` is not automatically eligible for calculation, and an `other` event can be eligible. Consumers MUST NOT infer a metric role from `kind`, **`label`**, `content`, or `text` alone. Unknown kinds MUST NOT cause rejection of the complete extension.

1.4.4 Dates, Ordering, and Duplicates

Producers MUST preserve the source value in `value` and normalize `timestamp` to UTC. A full date such as `2026-10-16` is represented as `2026-10-16T00:00:00Z` with `date_only: true`. This midnight is a sorting convention, not evidence that the event occurred at midnight. A source date-time at midnight retains `date_only: false` even when the calculation treats it as day precision.

The reference implementation truncates sub-millisecond precision in `timestamp` while retaining the original value. It also accepts unambiguous year-first slash dates and treats offset-free source date-times as UTC. Producers using these compatibility rules SHOULD document them; consumers MUST NOT reinterpret normalized timestamps in their own local timezone. Invalid or ambiguous source dates MUST NOT be fabricated into events or anchors.

Events MUST be sorted by normalized timestamp. Producers SHOULD resolve ties deterministically, for example by label and source. Array order MUST NOT imply causality or an order within a date-only day.

Producers MAY collapse duplicate extractions of the same source assertion, but SHOULD retain distinct provenance and MUST NOT merge different metric roles merely because timestamps coincide. Coincident dates from independent sources are not duplicate evidence by themselves.

1.5 Metric Evidence

Metrics are OPTIONAL. An event without a metric role remains useful timeline information. Producers assigning roles MUST establish both the event's meaning and its applicability to the requested subject.

1.5.1 Roles and Evidence Bases

`metric_role`	Meaning
`discovery`	Recorded discovery of the vulnerability.

metric_role	Meaning
report	Report to, receipt by, or notification of a vendor or coordinator.
acknowledgement	Acknowledgement of the report.
disclosure	Public disclosure or publication of information covering the vulnerability.
exploit_publication	Report of a published proof of concept or exploit.
exploitation	Recorded exploitation observation or qualifying exploitation evidence.
exploitation_report	A report or catalog listing asserting exploitation, without establishing its actual first occurrence.
patch	Recorded evidence that a corrective measure is available.
patch_observed	A sighting reporting a patched system, providing an upper bound on availability.

metric_basis	Meaning
recorded_event	Structured event type or recognized source field.
sighting_report	A typed sighting; its report date may lag the underlying activity.
kev_first_seen	KEV evidence treated as a first-seen observation.
kev_report	KEV assertion, catalog listing, import, or reported patch date.

Neither roles nor bases guarantee source accuracy. A proof of concept does not establish observed exploitation. A patched sighting does not establish deployment across all affected systems. A vendor fix date does not establish that every affected product or version has a fix.

1.5.2 Eligibility and Source Mapping

The reference implementation uses the following mappings where subject scope is established:

Input	Role	Basis
X-03 `discovered`	`discovery`	`recorded_event`
X-03 `reported`, `received`, `affected-party-notified`	`report`	`recorded_event`
X-03 `acknowledged`	`acknowledgement`	`recorded_event`
X-03 **`public`**`-disclosure`, `advisory-published`	`disclosure`	`recorded_event`
X-03 `exploit-observed`	`exploitation`	`recorded_event`
X-03 `fix-available`	`patch`	`recorded_event`
Recognized discovery, report, acknowledgement, publication, exploitation, or fix date fields	Corresponding role	`recorded_event`
CSAF vulnerability `release_date`	`disclosure`	`recorded_event`
CSAF remediation `date` with `category: vendor_fix`	`patch`	`recorded_event`
Sighting `exploited`	`exploitation`	`sighting_report`
Sighting `patched`	`patch_observed`	`sighting_report`
Sighting `published-proof-of-concept`	`exploit_publication`	`sighting_report`
Eligible KEV first-seen evidence	`exploitation`	`kev_first_seen`
Eligible KEV assertion, listing, or import date	`exploitation_report`	`kev_report`
KEV evidence `details.patchedSince`	`patch`	`kev_report`

Known field names qualify only in documented semantic locations, such as the record root, `cveMetadata`, `containers.cna`, `document.tracking`, or the matching CSAF vulnerability entry. A publication date nested in a reference, tool, generator, provider, or regulatory notification MUST NOT automatically become the subject's disclosure date. A CVE free-text timeline entry has no structured event type and MUST NOT acquire a role merely from its wording.

A linked record MUST explicitly describe the requested vulnerability or a known equivalent identifier before supplying its vulnerability-specific anchors. References, correlations, and generic GCVE relationships alone are insufficient. For a multi-vulnerability CSAF advisory, per-vulnerability dates MUST be taken from the matching vulnerability entry. When the requested subject is itself an advisory covering several vulnerabilities, producers MUST NOT combine different vulnerabilities' dates into a single vulnerability lifecycle.

Advisory publication can provide disclosure evidence for a vulnerability covered by that publication. If a later revision added the vulnerability, producers SHOULD use the date of the revision introducing it and set `added_in_revision: true`. The reference implementation recognizes selected revision descriptions for this purpose; incomplete revision histories and unrecognized wording remain limitations.

Only KEV entries currently asserting exploitation qualify as exploitation inputs. In the reference implementation, `status.exploited` must be **`true`** and status reasons `suspected`, `disputed`, and `withdrawn` are excluded. Missing status reasons or `confirmed`, `historical`, and `unknown` are accepted; the original status is retained so consumers can apply stricter criteria.

The reference implementation treats first-hand KEV evidence, or a first-seen date earlier than its assertion date, as `kev_first_seen`. Otherwise the listing, assertion, or import supplies `kev_report`. These are evidence classifications, not independent verification of the catalog's dates.

Catalog backlog dates SHOULD carry `batch_listing: true`. The reference implementation flags days with at least 25 exploited entries from the same catalog, separately considering catalog listing days and instance import days. This threshold is an implementation heuristic, not proof of when exploitation began. Its publication-copy heuristic flags KEV exploitation dates less than one second from a sub-day disclosure timestamp as `publication_copy: true`; such dates remain visible but MUST NOT supply exploitation anchors under that profile.

Reservation, ordinary record updates, EPSS predictions, remediation deadlines, `fix-developed`, `mitigation-available`, and planned disclosure MUST NOT substitute for the defined anchors. In particular, neither mitigation nor fix development establishes fix availability.

1.6 Duration Metrics

When `metrics` is present, it MUST include `as_of` and `anchors`. Producers MAY omit individual metrics; an omitted metric means it was not supplied.

`as_of` MUST be an RFC 3339 UTC timestamp identifying the calculation time. Future events MAY remain in `data` but MUST NOT become observed anchors for that calculation. Metrics describe the

collected evidence set and SHOULD NOT change merely because a user filters or sorts a displayed timeline.

1.6.1 Anchor Selection

`anchors` MUST contain the following members. Each value is a complete selected event object or **`null`** when no eligible event is available.

Symbol	Anchor	Selection
V	`discovery`	Earliest eligible discovery evidence.
R	`report`	Earliest eligible vendor or coordinator report evidence.
A	`acknowledgement`	Earliest eligible acknowledgement evidence.
D	`disclosure`	Earliest eligible disclosure evidence.
X	`exploit_publication`	Earliest eligible public exploit evidence.
E	`exploitation`	Earliest eligible `exploitation` or `exploitation_report` evidence.
P	`patch`	Earliest eligible `patch` or `patch_observed` evidence.
O	`origin`	Earliest of V, R, and D.
N	Calculation clock	`metrics.as_of`; not an `anchors` member.

Selection MUST compare normalized timestamps across eligible sources. At equal timestamps, `recorded_event` and `kev_first_seen` SHOULD take precedence over report bases; remaining ties SHOULD be resolved by source and field path. The selected event MUST retain its provenance, original date, precision, role, basis, and flags.

The earliest eligible evidence bounds what is recorded in the collection. It does not necessarily identify the actual first occurrence. For example, a distributor's discovery date may record when that distributor learned of an already public issue, and the earliest recorded fix may be downstream of an earlier upstream fix.

1.6.2 Defined Metrics

For the following formulas, Q is P when known, otherwise N. Only the three formulas explicitly using Q permit this calculation-clock fallback.

Metric	Formula	Interpretation
`vendor_response_time`	A − R	Report to acknowledgement.
`time_to_patch`	P − R	Report to patch evidence.
`total_time_to_patch`	P − V	Discovery to patch evidence.
`disclosure_to_patch`	P − D	Public disclosure to patch evidence.
`time_unpatched`	max(0, Q − O)	Recorded waiting interval, open if no patch is recorded.
`time_to_exploit`	E − O	Origin to exploitation evidence.
`disclosure_to_exploit`	E − D	Disclosure to exploitation evidence.
`exploitation_to_patch`	Q − E	Exploitation evidence to patch evidence, or an open interval.
`disclosure_to_exploit_publication`	X − D	Disclosure to public exploit evidence.
`exploit_publication_to_patch`	Q − X	Public exploit evidence to patch evidence, or an open interval.

The start of `time_to_patch` MUST NOT silently fall back from report to disclosure or discovery. Public exploit publication MUST NOT substitute for exploitation. Negative signed intervals MUST be retained except for `time_unpatched`, whose nominal value and bounds are clamped to zero. A negative exploitation-to-patch interval can indicate that patch evidence predates exploitation evidence; its precision and evidence bases must still be considered.

1.6.3 Metric Fields

Member	Type	Required	Description
`available`	boolean	yes	Whether both endpoints required by the formula are available.
`missing`	array of strings	when unavailable	Missing endpoint names: `start`, `end`, or both.
`seconds`	number	when available	Nominal signed interval in seconds, subject to the clamping rule.
`hours`	number	when available	`seconds / 3600`.
`days`	number	when available	`seconds / 86400`.
`min_seconds`	number	when available	Lower bound accounting for endpoint precision.
`max_seconds`	number	when available	Upper bound accounting for endpoint precision.
`approximate`	boolean	when available	Whether either evidence endpoint is treated as day precision.
`ongoing`	boolean	when available	Whether the calculation clock substitutes for missing patch evidence.
`order_known`	boolean	when available	Whether endpoint precision establishes their temporal order.

Member	Type	Required	Description
`end_before_start`	boolean	when available	Whether the end certainly precedes the start, before clamping.
`batch_listing`	boolean	no	Whether either selected endpoint carries the batch-listing flag.
`discovery_after_disclosure`	boolean	no	On `total_time_to_patch`, whether discovery certainly follows the disclosure day or instant.
`start`	object	when available	Complete selected start event.
`end`	object	when available	Complete selected end event, or the calculation-clock object.

An unavailable metric MUST omit numeric durations and endpoint objects rather than supplying fabricated zeros. For example:

```
{
  "vendor_response_time": {"available": false, "missing": ["start", "
     end"]},
  "time_to_patch": {"available": false, "missing": ["start"]}
}
```

When P is absent, the three Q-based metrics use `ongoing: true`, provided their start exists. Their end MUST be a synthetic clock object containing `timestamp` equal to `as_of`, `date_only: false`, `label: As of`, and `source: Calculation clock`. This clock object is not a source event and is not included in `data` or its counts. Other patch metrics remain unavailable.

No recorded patch does not establish that no patch exists. No exploitation evidence does not establish that exploitation has never occurred. These public evidence intervals MUST NOT be presented as an organization's actual exposure period without deployment evidence for that organization.

1.6.4 Precision and Bounds

For compatibility with the reference implementation, a calculation evidence endpoint has day precision if `date_only` is **`true`** or its normalized timestamp is exactly midnight UTC. This conservative midnight rule also applies when `date_only` is **`false`**. It MUST NOT be applied to the synthetic calculation clock.

Let s and e be the normalized endpoints. Let s_upper be s plus one day for a day-precision start, otherwise s; define e_upper similarly. Before clamping:

```
seconds     = e - s
min_seconds = e - s_upper
max_seconds = e_upper - s
```

All differences are in seconds. A day is 86400 seconds for these calculations. The bounds conservatively cover uncertainty within the UTC day, not uncertainty about the truth of the source assertion. Both exact endpoints produce identical nominal, minimum, and maximum values. Units SHOULD be exported without presentation rounding; displayed values MAY be rounded.

`end_before_start` is true when an exact end is earlier than s, or when a day-precision end's e_upper is at or before s. `order_known` is true when that condition holds or e is at or after s_upper. Otherwise, consumers MUST NOT assert an order based solely on the nominal duration. Flags are calculated before `time_unpatched` clamping.

For example, exploitation at `2026-10-05T10:00:00Z` and a patch dated `2026-10-05` yield a nominal exploitation-to-patch interval of −10 hours with bounds of −10 to +14 hours. The metric is approximate and its order is unknown.

If the selected discovery certainly follows public disclosure, `total_time_to_patch` SHOULD carry `discovery_after_disclosure: true`. This warns that the recorded discovery may describe a publisher learning of the issue. It does not change the selected origin or silently replace the discovery anchor.

1.7 Example Timeline

The following synthetic standalone payload illustrates publication, a date-only fix, a linked advisory, and a sighting. Identifiers and sources are illustrative; they do not assert a real vulnerability history.

```
{
  "metadata": {
    "vulnerability": "GCVE-1-2026-12345",
    "count": 3,
    "kinds": {"milestone": 1, "sighting": 1, "kev": 0, "other": 1},
```

```
    "linked_omitted": 0
  },
  "data": [
    {
      "timestamp": "2026-10-03T09:00:00Z",
      "date_only": false,
      "value": "2026-10-03T11:00:00+02:00",
      "kind": "milestone",
      "label": "Published",
      "source": "Example GNA",
      "record": "GCVE-1-2026-12345",
      "field": "cveMetadata.datePublished",
      "metric_role": "disclosure",
      "metric_basis": "recorded_event"
    },
    {
      "timestamp": "2026-10-05T00:00:00Z",
      "date_only": true,
      "value": "2026-10-05",
      "kind": "other",
      "label": "Vendor fix",
      "source": "Example vendor CSAF",
      "record": "EXAMPLE-SA-2026-001",
      "field": "vulnerabilities.0.remediations.0.date",
      "linked_record": true,
      "metric_role": "patch",
      "metric_basis": "recorded_event"
    },
    {
      "timestamp": "2026-10-05T10:00:00Z",
      "date_only": false,
      "value": "2026-10-05T10:00:00Z",
      "kind": "sighting",
      "label": "Exploited",
      "source": "https://example.org/observations/1",
      "record": "GCVE-1-2026-12345",
      "field": "creation_timestamp",
      "uuid": "b49cb7c6-40a5-4e8f-847e-51e833c979d2",
      "metric_role": "exploitation",
      "metric_basis": "sighting_report"
    }
  ]
}
```


This payload omits optional `metrics`. At a calculation time of `2026-10-06T09:00:00Z`, its anchors would have V, R, A, and X set to **`null`**, D and O equal to the publication event, P equal to the fix event, and E equal to the sighting. Selected results are:

Metric	Nominal result	Precision or availability
`disclosure_to_exploit`	49 hours	Exact timestamp difference; the sighting remains report-based evidence.
`disclosure_to_patch`	39 hours	Approximate; bounds 39 to 63 hours.
`time_unpatched`	39 hours	Approximate; bounds 39 to 63 hours, `ongoing: false`.
`exploitation_to_patch`	−10 hours	Approximate; bounds −10 to +14 hours, `order_known: false`.
`vendor_response_time`	Unavailable	Both report and acknowledgement are missing.
`time_to_patch`	Unavailable	Report is missing.
`total_time_to_patch`	Unavailable	Discovery is missing.

An available metric retains the full selected events. For example, a calculation of `exploitation_to_patch` on this payload MUST copy `data[2]` into `start` and `data[1]` into `end`, and include the numeric values and flags described above.

1.8 Processing and Validation Requirements

1. Producers MUST validate required members, types, date normalization, and count consistency before publication.
2. Producers MUST keep event provenance and MUST distinguish displayed source dates from eligible metric inputs.
3. Consumers MUST NOT assume that every event in a linked advisory concerns the requested vulnerability.
4. Metric endpoints MUST follow the defined formulas and eligibility rules. Source endpoints MUST correspond to events in `data`; clock endpoints are the stated exception. Copies in anchors and metrics MUST remain consistent with those events.
5. All numeric durations MUST be finite. `min_seconds` MUST be no greater than `seconds`, which MUST be no greater than `max_seconds` after applicable clamping.

6. Unknown members, kinds, roles, or evidence bases MUST NOT cause rejection of the complete extension. Consumers MUST NOT use unknown roles or bases in calculations unless their semantics are understood.
7. Consumers SHOULD preserve unknown members when forwarding the extension.
8. Consumers MUST distinguish omission, `null` anchors, unavailable metrics, and measured zero durations.
9. A producer recomputing metrics MUST update `as_of`, anchor selection, endpoint copies, and durations consistently. UI filtering MUST NOT silently change the collection over which exported metrics were calculated.

1.9 Updates and Interoperability

An aggregate timeline is a snapshot. New sources, corrected dates, withdrawn assertions, changed links, and collection limits can change both events and selected anchors. Producers SHOULD retain sufficient source history to explain material changes. An earlier snapshot's metrics MUST NOT be combined with a later snapshot's event set as if they were calculated together.

When collecting records that already contain X-04, producers SHOULD avoid recursively harvesting the embedded aggregate's event copies and calculated endpoints as new source assertions. This prevents feedback and duplicate evidence.

This extension complements:

- BCP-02, for vulnerability handling and disclosure practice.
- BCP-03, for publication and synchronization. Historical event timestamps and `metrics.as_of` do not replace synchronization timestamps.
- BCP-05, for the containing GCVE record.
- BCP-05-X-03, for typed handling events used as inputs.
- BCP-07, for the full KEV assertions underlying timeline dates.
- BCP-12, for the full sighting records underlying timeline dates.

Metrics borrow lifecycle concepts used by the Vulnerability Timeline and Exposure Metrics (VTEM) framework, as described in the reference implementation's documentation. This extension does not claim to implement every VTEM date or an organization-specific deployment or exposure model. The event projection does not replace the original vulnerability, advisory, KEV, or sighting records.

1.10 Privacy, Security, and Trust Considerations

Aggregated timelines can expose confidential coordination dates, personal data, internal identifiers, and source content. Publishers MUST review the payload for its intended distribution and MUST remove secrets and inappropriate private data. Optional `content` and `text` SHOULD be omitted when unnecessary.

Consumers MUST treat source strings and quoted content as untrusted data and MUST NOT execute instructions embedded in them. Renderers SHOULD escape source text and MUST validate URLs before making them actionable. A URL MUST NOT trigger an automatic fetch without appropriate controls against unsafe destinations.

An event count is not an independent-source count: many sightings can repeat one claim. Neither a large timeline nor precise arithmetic establishes source reliability. Consumers SHOULD display provenance, evidence bases, missing data, approximation, unknown ordering, backlog flags, and collection omissions alongside derived measurements. Vendor or product comparisons require comparable collection coverage and evidence scope.

1.11 Example JSON Paths

The aggregate timeline in a CVE record carrying a GCVE extension is located at:

```
containers.cna.x_gcve[0].extensions.bcp-05-x-04.x_timeline
```

Its event list and optional calculated durations are located at:

```
containers.cna.x_gcve[0].extensions.bcp-05-x-04.x_timeline.data
containers.cna.x_gcve[0].extensions.bcp-05-x-04.x_timeline.metrics
```

1.12 Reference Implementation

This draft is based on a sample timeline JSON export and the following Vulnerability Lookup sources, pinned to commit `038227307a41574d05ad9e48ea0a84f2110ff33c`:

- Timeline collection and calculation implementation.
- Evidence selection, duration semantics, and implementation limitations.

The implementation exposes standalone timelines at `GET /api/vulnerability/<id>/timeline` and `/vuln/<id>/timeline.json`. Its collection budgets, extraction rules, and

heuristics can evolve; consumers requiring reproducible studies SHOULD retain the exported snapshot and the generator version used to produce it.