Network Working Group
Internet-Draft
Intended status: Informational
Expires: 30 March 2026

L.J. Reilly
Independent Submission

26 September 2025

Reilly Sentinel Protocol (RSP): Blockchain-Anchored Integrity for AI Datasets, Training, Fine-Tuning, and Inference Provenance

draft-reilly-sentinel-protocol-00

Abstract

The Reilly Sentinel Protocol (RSP) specifies an interoperable, dual-layer method for establishing integrity, provenance, and auditability across the artificial intelligence (AI) lifecycle. RSP defines a Sentinel Evidence Package (SEP) that binds payload digests, provenance metadata, signatures, blockchain timestamp proofs, and resolvable identifiers (DOIs). This enables tamper-evident, independently verifiable receipts for datasets, data transformations, training jobs, checkpoints, fine-tuning runs, evaluations, and inference outputs.

RSP is transport-agnostic and serializable in JSON and CBOR. It leverages existing IETF and industry building blocks, including COSE/CMS signatures, CBOR (RFC 8949), CDDL (RFC 8610), JSON (RFC 8259), and NTS-secured time (RFC 8915). Anchoring is done via append-only blockchain receipts (e.g., OpenTimestamps-style proofs) and identity/lineage is stabilized with a DOI registry. The result is an evidence-grade audit trail for regulated and safety-critical AI deployments.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 30 March 2026.

Table of Contents

1. Introduction

Artificial Intelligence (AI) systems are increasingly used in high-stakes contexts (defense, healthcare, finance, critical infrastructure). Confidence in AI outcomes depends on the ability to show where data came from, how models were trained or adapted, when inferences were produced, and whether any of these artifacts have been altered.

RSP addresses these needs by defining a minimal yet extensible evidence container -- the Sentinel Evidence Package (SEP) -- and a verification process that any independent party can execute without trusting the producer's infrastructure. RSP is content- and model-agnostic: it does not dictate the model architecture or task; it only standardizes how integrity, time, identity, and lineage are recorded and verified.

RSP combines two complementary layers of permanence:

* Blockchain anchoring: an append-only proof of existence by a given time for payload digests and roll-up digests.

* DOI registration: globally resolvable, citable identifiers with metadata, lineage, and retention semantics.

This dual anchoring makes later manipulations detectable and enables reliable long-term discovery and citation.

2. Conventions and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 (RFC 2119) (RFC 8174) when, and only when, they appear in all capitals, as shown here.

SEP: Sentinel Evidence Package; the container defined by this

document.

Artifact: Any AI lifecycle asset -- dataset, data transform, training configuration, training log, model checkpoint, fine-tuning diff, evaluation artifact, or inference record.

Digest: A cryptographic hash (e.g., SHA-256) of a payload or Merkle root over chunked payloads.

OTS: OpenTimestamps-style proof object or functionally equivalent blockchain timestamp receipt.

DOI: Digital Object Identifier; a resolvable identifier (via DataCite/Crossref or internal Handle) providing metadata and lineage stability.

COSE: CBOR Object Signing and Encryption; RSP uses COSE signatures (RFC 9052, RFC 9053).

CMS: Cryptographic Message Syntax (RFC 5652).

3. Architecture and Roles

Producer: Creates artifacts, computes digests, collects device/environment attestations, signs SEPs, and requests anchoring and DOI registration.

Registrar: Provides two sub-services: (a) Anchoring Orchestrator that submits digests to public chains and returns proofs; (b) DOI Registrar that mints DOIs and stores metadata, lineage, and fixity.

Verifier: Recomputes digests, validates signatures, checks OTS receipts against independent nodes, and resolves DOI metadata and lineage without trusting Producer or Registrar beyond their signatures.

Storage: WORM (Write-Once-Read-Many) or equivalent immutable object storage for SEPs, proofs, and payloads under retention policy.

Time: Authenticated time sources (NTS, RFC 8915) and monotonic counters to resist back-dating and rollback.

RSP does not mandate a specific trust relation among these roles; in many deployments, Producer and Registrar are separate entities.

4. Sentinel Evidence Package (SEP) Data Model

An SEP is a self-describing manifest that binds digest(s), metadata, signatures, proofs, and identifiers. The canonical fields are:

* version: Protocol version string (e.g., "1.0").

* artifact.type: "dataset", "datatransform", "training.config", "training.log", "training.checkpoint", "fine-tune.diff", "evaluation", "inference", or an extension token.

* payloads[]: Optional array with entries { cid, size, mime, chunking }; payload storage MAY be external. 'cid' MUST encode algorithm and value (e.g., "sha256:HEX").

* digests: Object with at least sha256 and sha3_256 members (hex-encoded). For chunked payloads, digests.root is the Merkle root, and digests.leaves MAY be omitted.

* timestamps: wallclock (RFC 3339) and monotonic counter. The 'source' SHOULD indicate NTS, Roughtime, or equivalent.

* authors[]: Optional list of creators/operators; MAY include device IDs and certificate references.

* attestation: Optional device/firmware/build attestation claims (out of scope to specify here; see RATS, RFC 9334).

* rem: Anchoring and identity sub-object:
  - rem.ots_proof_ref: URN or locator to proof file.
  - rem.doi: DOI string.
  - rem.lineage: parent[], siblings[], supersedes[].

* class: Optional classification/handling indicators (tokenized).

* policies: Retention and access policy references.

* signatures[]: One or more COSE_Sign1 or CMS signatures covering the manifest canonical form. Certificate chains MUST be included.

The manifest MUST be canonicalized prior to signing; canonical JSON (JCS, RFC 8785) or deterministic CBOR (RFC 8949 Section 4.2) MAY be used.

5. Serialization and Media Types

RSP SEPs MAY be emitted as JSON (RFC 8259) or CBOR (RFC 8949). This document defines two media types for IANA registration:

* application/rsp-ep+json
* application/rsp-ep+cbor

The "+" structured suffix follows RFC 6839. See Section 14 for IANA templates.

6. Anchoring and Proofs

Producers submit digests (payload or roll-up Merkle roots) to the Anchoring Orchestrator. The orchestrator MUST anchor to at least one public append-only chain and SHOULD support multiple chains for diversity. Proofs MUST be exportable and independently checkable.

OpenTimestamps receipts are RECOMMENDED where applicable. When using other schemes, a proof MUST include sufficient data for independent verification (e.g., chain identifier, transaction locator, inclusion path).

Roll-up anchoring: Operators SHOULD periodically compute a Merkle tree over all new artifact digests and anchor the root to reduce per-artifact anchoring cost and to provide compact audit coverage.

7. DOI Registration and Metadata

After anchoring, the Registrar mints a DOI and stores a metadata record containing at least: title or short label, creators, creation time, fixity (digests), lineage links, and policy URIs. DOIs for classified material MAY be internal Handles resolvable on private networks; unclassified artifacts MAY use public DOIs (e.g., DataCite).

A DOI SHOULD be minted for each material evolution that changes fixity (e.g., new dataset version, fine-tune result, evaluation report). Prior DOIs MUST NOT be deleted; instead, metadata MAY indicate "superseded-by".

8. Protocol Operations

The following abstract operations are defined; concrete APIs are out of scope, but typical deployments use REST or gRPC.

8.1 SEAL

Inputs: manifest draft (unsigned), digests, optional attestation.
Action: Canonicalize, sign (COSE/CMS), return SEP (unsigned -> signed).

8.2 ANCHOR

Inputs: digest(s) (payload or roll-up). Action: Anchor to one or more chains; return proof reference(s).

8.3 REGISTER

Inputs: metadata (including fixity, lineage). Action: Mint DOI, bind DOI <-> SEP <-> proofs; return DOI.

8.4 VERIFY

Inputs: DOI or SEP URI (plus payload, if accessible). Action: Recompute digests; validate signatures; check time discipline; verify proofs; emit Verification Report.

9. Verification Algorithm

Given a DOI or SEP:

1. Resolve DOI (if provided) to retrieve minimal metadata including fixity, lineage, and pointers to SEP and payload (subject to policy).

2. Fetch SEP. Parse and canonicalize the manifest according to its serialization; verify that the 'digests' field matches the payload (if accessible).

3. Validate signatures (COSE/CMS). Check certificate validity and revocation (OCSP/CRL). Attestation claims MAY be evaluated via a verifier (cf. RATS, RFC 9334).

4. Validate time: ensure wallclock is plausible (NTS/secure time) and monotonic counter continuity holds relative to adjacent SEPs.

5. Validate anchoring proofs independently using public chain data.

6. Emit a Verification Report with fields: hash_ok, sig_ok, attestation_ok, ots_ok, time_ok, doi_ok, and overall verdict. The report SHOULD itself be signed.

10. Error Handling

RSP defines abstract error conditions with suggested codes/strings:

ERR_PARSE - malformed SEP or encoding

```
ERR_DIGEST_MISMATCH- recomputed digest != manifest
ERR_SIG_INVALID    - signature or chain invalid
ERR_TIME_INVALID   - unauthenticated or non-monotonic time
ERR_PROOF_INVALID  - proof not verifiable on stated chain
ERR_DOI_RESOLVE    - DOI metadata unavailable/inconsistent
ERR_POLICY_DENIED  - access denied by policy
```

Implementations SHOULD map these to transport-specific error responses.

11. Manageability and Telemetry

Deployments SHOULD monitor:

- Anchoring SLA (time to first chain inclusion).
- Coverage (% artifacts with valid SEPs and DOIs).
- Verification pass rate and dominant failure reasons.
- Key health (expiration, revocation events).
- Time-source health (NTS/stratum, Roughtime reachability).

Daily or mission-rollup Merkle roots SHOULD be recorded and anchored, enabling compact audits.

12. Privacy Considerations

SEPs record integrity metadata; payloads MAY be withheld. Where privacy is required, field-level encryption SHOULD be used for sensitive manifest attributes while keeping digests and proofs public. DOI records for sensitive artifacts SHOULD minimize personally identifiable information. Policy URIs SHOULD reflect access constraints.

13. Security Considerations

Key compromise: Private keys used for COSE/CMS MUST be protected in HSMs or equivalent. Short-lived device certificates and revocation (OCSP/CRL) are RECOMMENDED. Compromised keys MUST NOT trigger data deletion; instead, append revocation facts and supersede with new signatures.

Time attacks: Producers MUST use authenticated time (NTS) and monotonic counters. Multiple time sources are RECOMMENDED. Large backward jumps MUST be flagged.

Proof stability: Anchoring to multiple independent chains (heterogeneous consensus) reduces correlation risk. Re-anchoring of historical Merkle roots over time is RECOMMENDED for longevity.

Replay/substitution: Bind signatures to digests and policy context; DOIs provide stable identity to detect swaps.

Supply chain: Model and dataset supply chain attestations (e.g., SBOM-like metadata, RATS evidence) are RECOMMENDED but out of scope to normatively specify here.

Security considerations follow RFC 3552 guidance.

14. IANA Considerations

IANA is requested to register the following media types:

14.1 application/rsp-ep+json

```
Type name:           application
Subtype name:        rsp-ep+json
Required parameters: none
Optional parameters: charset (per RFC 6838)
Encoding considerations:  binary; JSON (RFC 8259)
Security considerations:  see Section 13
Interoperability considerations:  none
Published specification:  this document
Applications that use this media type:  AI provenance systems
Fragment identifier considerations:  per RFC 7303
Additional information:
  Magic number(s):  n/a
  File extension(s):  .rsp.json
  Macintosh file type code(s):  n/a
Person & email address to contact for further information:
  Lawrence John Reilly Jr. <lawrencejohnreilly@gmail.com>
Intended usage:  COMMON
Restrictions on usage:  none
Author:  IETF
Change controller:  IESG
```

14.2 application/rsp-ep+cbor

```
Type name:           application
Subtype name:        rsp-ep+cbor
Required parameters: none
Optional parameters: none
Encoding considerations:  binary; CBOR (RFC 8949)
Security considerations:  see Section 13
Interoperability considerations:  none
Published specification:  this document
Applications that use this media type:  AI provenance systems
Fragment identifier considerations:  none
Additional information:
  Magic number(s):  n/a
  File extension(s):  .rsp.cbor
  Macintosh file type code(s):  n/a
Person & email address to contact for further information:
  Lawrence John Reilly Jr. <lawrencejohnreilly@gmail.com>
Intended usage:  COMMON
Restrictions on usage:  none
Author:  IETF
Change controller:  IESG
```

15. Implementation Status

A background report describing RSP’s intent and design is archived at Zenodo (doi:10.5281/zenodo.17103522). Reference implementations for SEP creation, anchoring integration, and verification are planned for public release.

16. References

16.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, May 2017.

[RFC8259] Bray, T., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 8259, December 2017.

[RFC8610] Birkholz, H., Vigano, C., and C. Bormann, "Concise Data

Definition Language (CDDL): A Notational Convention to Express CBOR and JSON Data Structures", RFC 8610, June 2019.

[RFC8785] rfc-editor.org, "JSON Canonicalization Scheme (JCS)", RFC 8785, June 2020.

[RFC8915] Franke, D., "Network Time Security for the Network Time Protocol", RFC 8915, September 2020.

[RFC8949] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", STD 94, RFC 8949, December 2020.

[RFC9052] Schaad, J., "CBOR Object Signing and Encryption (COSE): Structures and Process", RFC 9052, August 2022.

[RFC9053] Schaad, J., "CBOR Object Signing and Encryption (COSE): Algorithms", RFC 9053, August 2022.

[RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, July 2002.

16.2. Informative References

[RSP] Reilly, L. J., "Reilly Sentinel Protocol", Zenodo, doi:10.5281/zenodo.17103522.

[REM] Reilly, L. J., "Reilly EternaMark (REM) Protocol", Zenodo, doi:10.5281/zenodo.17129012.

[RATS] Birkholz, H., et al., "Remote Attestation Procedures Architecture", RFC 9334, November 2022.

[CMS] Housley, R., "Cryptographic Message Syntax (CMS)", RFC 5652, September 2009.

[SCITT] IETF, "Supply Chain Integrity, Transparency, and Trust (SCITT) Working Group Charter", IETF Datatracker.

[C2PA] Coalition for Content Provenance and Authenticity, "C2PA Specification", 2023.

[OTS] OpenTimestamps community, "OpenTimestamps: Scalable Timestamping", online documentation.

[DATACITE] DataCite Metadata Schema 4.x, DataCite.

Appendix A. CDDL for CBOR SEP

```
sep = {
  version: tstr,
  artifact: {
    type: tstr
  },
  ? payloads: [* payload],
  digests: {
    sha256: tstr,        ; lowercase hex
    sha3_256: tstr,
    ? root: tstr
  },
  timestamps: {
    wallclock: tstr,     ; RFC3339
    ? monotonic: uint,
    ? source: tstr
  },
  ? authors: [* { name: tstr, ? id: tstr }],
```

```
  ? attestation: any,
  rem: {
    ots_proof_ref: tstr,
    doi: tstr,
    lineage: {
      ? parent: [* tstr],
      ? siblings: [* tstr],
      ? supersedes: [* tstr]
    }
  },
  ? class: any,
  ? policies: any,
  signatures: [* any]    ; COSE_Sign1 or CMS
}

payload = { cid: tstr, size: uint, ? mime: tstr, ? chunking: tstr }
```

Appendix B. Example JSON SEP

```
{
  "version": "1.0",
  "artifact": {"type": "training.checkpoint"},
  "digests": {"sha256": "ab..", "sha3_256": "cd.."},
  "timestamps": {"wallclock": "2025-09-26T12:34:56Z", "source": "NTS"},
  "rem": {
    "ots_proof_ref": "urn:ots:btc:txid:...",
    "doi": "doi:10.rsp/abc123",
    "lineage": {"parent": ["doi:10.rsp/dataset456"]}
  },
  "signatures": [{"...": "COSE_Sign1 bytes (base64url)"}]
}
```

Appendix C. Verification Report (JSON)

```
{
  "doi": "doi:10.rsp/abc123",
  "hash_ok": true,
  "sig_ok": true,
  "attestation_ok": true,
  "ots_ok": true,
  "time_ok": true,
  "doi_ok": true,
  "verified_at": "2025-09-26T20:10:22Z"
}
```

Appendix D. Example OpenTimestamps Receipt

(Non-normative) An OTS receipt includes commitment operations and attestations that enable independent reconstruction of inclusion in a public chain. See [OTS].

Appendix E. Example DOI Metadata Mapping

```
title: "Model Checkpoint R1 (ResNet-50 finetune)"
creators: [{"name": "Org Unit 7"}]
created: "2025-09-26T12:34:56Z"
fixity: {"sha256": "...", "sha3_256": "..."}
lineage.parent: ["doi:10.rsp/dataset456"]
policy_uri: "abac://policy/default/retention/75y"
```

Acknowledgments

The author thanks reviewers and implementers who provided feedback on RSP design and alignment with REM and SCITT efforts.

Author’s Address

Lawrence John Reilly Jr.
Independent Submission
Email: lawrencejohnreilly@gmail.com

Expires: 30 March 2026