Network Working Group
Internet-Draft
Intended status: Experimental
Expires: 28 August 2026

S. Yamakawa, Ed.
AILEX LLC / VeritasChain Standards Organization

24 February 2026

Verifiable AI Provenance (VAP) Framework and Legal AI Profile (LAP)

draft-ailex-vap-legal-ai-provenance-02

Abstract

This document specifies the Verifiable AI Provenance (VAP) Framework, a cross-domain upper framework for cryptographically verifiable decision audit trails in high-risk AI systems, along with the Legal AI Profile (LAP), a domain-specific instantiation for legal AI and LegalTech systems.

VAP defines common infrastructure including hash chain integrity, digital signatures, unified conformance levels (Bronze/Silver/Gold), external anchoring via RFC 3161 Time-Stamp Protocol and compatible transparency services (including IETF SCITT), a Completeness Invariant pattern guaranteeing no selective logging, standardized Evidence Pack format for regulatory submission, and privacy-preserving verification protocols.

LAP extends VAP for the judicial AI domain, addressing unique requirements including attorney oversight verification (Human Override Coverage), three-pipeline completeness invariants for legal consultation, document generation, and fact-checking, tiered content retention with legal hold protocols for judicial discovery compliance, graduated override enforcement mechanisms, and privacy-preserving fields designed to maintain attorney-client privilege while enabling third-party auditability.

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 28 August 2026.

Table of Contents

1. Introduction

The deployment of AI systems in high-risk domains -- including finance, healthcare, transportation, and the administration of justice -- creates a structural accountability gap. AI decisions that affect fundamental rights and societal infrastructure lack standardized, cryptographically verifiable audit trails that independent third parties can inspect.

Current approaches rely on trust-based governance: AI providers assert that their systems are safe and well-logged, but no independent party can cryptographically verify these claims. The Verifiable AI Provenance (VAP) Framework addresses this gap by defining a "Verify, Don't Trust" architecture for AI decision provenance.

This document defines two complementary specifications:

1. VAP Framework (Part I): A cross-domain upper framework defining common infrastructure for verifiable AI provenance applicable to any high-risk AI domain.

2. Legal AI Profile (LAP) (Part II): A domain-specific profile for legal AI systems, addressing requirements arising from professional regulation of attorneys and high-risk AI system governance.

1.1. Scope

VAP targets AI systems where "system failure could cause significant and irreversible harm to human life, societal infrastructure, or democratic institutions." This intentionally strict scope distinguishes VAP from general-purpose logging frameworks.

LAP specifically addresses legal AI systems that provide AI-powered legal consultation, document generation, and fact-checking services to licensed attorneys.

1.2. Design Philosophy

The core principle is "Verify, Don’t Trust." Rather than relying on AI providers’ claims about the safety and integrity of their systems, VAP enables independent, cryptographic verification of every AI decision’s provenance, completeness, and human oversight.

NOTE: This Internet-Draft is the authoritative specification for the VAP Framework and LAP Profile. Where differences exist between this document and other published descriptions of the VAP Framework (e.g., companion white papers or implementation guides), this Internet-Draft takes precedence.

2. Conventions and Definitions

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 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2.1. Terminology

VAP Verifiable AI Provenance Framework - the cross-domain upper framework defined in this document.

Profile A domain-specific instantiation of VAP (e.g., VCP for finance, CAP for content, LAP for legal).

LAP Legal AI Profile - the judicial AI domain profile defined in this document.

Provenance Cryptographically verifiable record of data origin, derivation, and history.

Completeness Invariant A mathematical guarantee that every attempt event has exactly one corresponding outcome event.

Evidence Pack A self-contained, signed package of provenance events suitable for regulatory submission and third-party audit.

External Anchor Registration of a Merkle root hash with an external trusted timestamping service such as [RFC3161] or a compatible transparency log such as IETF SCITT [IETF-SCITT].

Human Override An event recording a human professional’s review, approval, modification, or rejection of an AI-generated output.

Override Coverage The ratio of AI outputs reviewed by a human professional to total AI outputs, expressed as a percentage.

Causal Link A reference from an outcome event to its originating attempt event, establishing referential integrity within a pipeline.

Content Retention Tier One of three levels of content preservation (Full Content, Recoverable Hash, Hash-Only) defining the availability of original event content at a given point in the retention lifecycle.

Legal Hold A directive freezing the current retention tier for events within scope, preventing content deletion or tier transition during litigation, investigation, or regulatory inquiry.

Override Enforcement Level One of four graduated control levels (Metric Only, Warn, Gate, Strict) governing how the system responds when AI outputs lack corresponding HUMAN_OVERRIDE events.

Hash Input The canonical byte sequence over which EventHash is computed, defined as the JCS-canonicalized event with the security.event_hash, security.signature, and security.sign_algo fields removed. See Section 4.2.

3. VAP Framework Architecture

3.1. Layer Model

VAP is organized into four core layers, a common infrastructure layer, and a domain profile layer:

Integrity Layer Hash chain, digital signatures, timestamps (REQUIRED for all levels).

Provenance Layer Actor, input, context, action, and outcome recording (REQUIRED).

Accountability Layer Operator identification, approval chain, delegation records (REQUIRED for operator_id; RECOMMENDED for approval chain).

Traceability Layer Trace IDs, causal links, cross-profile references (REQUIRED for trace_id; OPTIONAL for cross-references).

Common Infrastructure Conformance levels, external anchoring,

completeness invariant, evidence packs, privacy-preserving verification, retention framework (availability depends on conformance level).

Domain Profile Layer Domain-specific event types, data model extensions, regulatory mappings (defined per profile).

3.2. Domain Profiles

VAP supports multiple domain profiles. Each profile MUST define:

1. Event Types: Domain-specific event type taxonomy.

2. Data Model Extensions: Additional fields beyond the VAP common event structure.

3. Conformance Mapping: Mapping to VAP Bronze/Silver/Gold levels.

4. Regulatory Alignment: Mapping to applicable regulations (informative).

5. Completeness Invariant Application: How the completeness invariant applies to domain-specific event flows.

Registered profiles include VCP (Finance), CAP (Content/Creative AI), and LAP (Legal AI, defined in Part II of this document). Additional profiles for automotive (DVP), medical (MAP), and public administration (PAP) domains are under development.

4. Cryptographic Foundation

4.1. Algorithm Requirements

All VAP-conformant implementations MUST support the primary algorithms listed below. Implementations SHOULD support at least one alternative algorithm in each category for migration purposes. Post-quantum algorithms are listed for future migration readiness and are currently OPTIONAL.

Category	Primary (MTI)	Alternative	Post-Quantum (Future MTI)
Hash	SHA-256	SHA-384, SHA-512	SHA3-256
Signature	Ed25519 ([RFC8032])	ECDSA P-256 ([RFC6979])	ML-DSA-65 ([FIPS204])
Encryption	AES-256-GCM	ChaCha20-Poly1305	N/A (see KEM row)
KEM	N/A (classical key exchange)	N/A	ML-KEM-1024 ([FIPS203])

Table 1: Required Cryptographic Algorithms

Note: The Post-Quantum signature algorithm ML-DSA-65 (formerly known as CRYSTALS-Dilithium, renamed upon FIPS 204 standardization in August 2024) provides security equivalent to AES-192. This parameter set was selected as a balance between signature size (3,309 bytes) and security margin for legal audit trails requiring multi-decade retention. ML-DSA-44 (AES-128 equivalent) was considered insufficient for the intended retention periods; ML-DSA-87 (AES-256 equivalent) imposes significantly larger signatures (4,627 bytes) with marginal practical benefit for this use case.

Note: ML-KEM-1024 (formerly known as CRYSTALS-Kyber, renamed upon FIPS 203 standardization in August 2024) is a key encapsulation mechanism (KEM), not an encryption algorithm. It is listed separately from symmetric encryption because KEM and symmetric encryption serve different roles: KEM establishes shared secrets for key agreement, while AES-256-GCM provides authenticated encryption of content.

Implementations MUST include algorithm identifiers in all cryptographic fields. The following table defines the canonical algorithm identifiers used in field values and wire-format prefixes:

Algorithm	Field Value (hash_algo / sign_algo)	Wire Prefix (in encoded strings)	Reference
SHA-256	"sha-256"	"sha-256:"	NIST FIPS 180-4
SHA-384	"sha-384"	"sha-384:"	NIST FIPS 180-4
SHA-512	"sha-512"	"sha-512:"	NIST FIPS 180-4
SHA3-256	"sha3-256"	"sha3-256:"	NIST FIPS 202
Ed25519	"ed25519"	"ed25519:"	[RFC8032]
ECDSA P-256	"ecdsa-p256"	"ecdsa-p256:"	[RFC6979]
ML-DSA-65	"ml-dsa-65"	"ml-dsa-65:"	[FIPS204]

Table 2: Canonical Algorithm Identifiers

All algorithm identifiers MUST be lowercase ASCII strings with hyphens as separators. The field value and wire prefix use the same string (the wire prefix appends a colon as delimiter). Implementations MUST perform case-insensitive comparison when validating algorithm identifiers but MUST produce lowercase output. These identifiers enable crypto agility and algorithm migration as specified in Section 4.4.

4.2. Hash Chain Specification

Events MUST be linked in a hash chain where each event’s hash includes the hash of the preceding event.

The Hash Input is computed by removing the following fields from the event before canonicalization:

* security.event_hash

* security.signature

The remaining fields (including security.hash_algo, security.sign_algo, and security.signer_id) are retained in the hash input. This explicit exclusion list prevents circular references (event_hash cannot be included in its own computation) and ensures that implementations agree on the hash input.

The computation proceeds as follows:

```
HashInput[n] = Event[n] with {security.event_hash,
                              security.signature} removed

EventHash[n] = HashAlgo(JCS-Canonicalize(HashInput[n]))

where:
  Event[n].header.prev_hash = EventHash[n-1]
  Event[0].header.prev_hash = null  (genesis event)
  HashAlgo is the algorithm specified in security.hash_algo
  JCS-Canonicalize follows RFC 8785
```

Canonicalization MUST follow [RFC8785] (JSON Canonicalization Scheme).

Chain integrity verification MUST confirm:

1. Each event's EventHash matches the recomputed hash over the Hash Input.

2. Each event's prev_hash matches the preceding event's EventHash.

3. The genesis event has a null prev_hash.

4. The hash_algo specified in each event is a supported algorithm.

4.3. Digital Signature Requirements

Every event MUST be signed. The signature MUST be computed over the EventHash bytes (not over the raw event):

```
Signature = SignAlgo.Sign(PrivateKey, EventHash_bytes)
```

The signature MUST be encoded as follows:

```
security.signature = sign_algo_id ":" Base64url(Signature_bytes)
```

where sign_algo_id is the canonical algorithm identifier from Table 2 matching the security.sign_algo field (e.g., "ed25519", "ecdsa-p256"), and Base64url encoding follows [RFC4648] Section 5 (URL-safe alphabet, no padding).

The primary mandatory-to-implement (MTI) signature algorithm is Ed25519 [RFC8032]. Implementations MUST support Ed25519 and SHOULD support at least one additional algorithm from Table 1 for migration purposes.

4.4. Algorithm Migration

VAP is designed for crypto agility per BCP 201 [RFC7696]. Algorithm migration proceeds as follows:

1. A new algorithm is added to the supported set via a specification update.

2. Implementations begin producing events with the new algorithm alongside the existing algorithm (dual-signing period, RECOMMENDED minimum 6 months).

3. Once all verifiers support the new algorithm, the old algorithm is deprecated.

4. After a deprecation notice period (RECOMMENDED minimum 12 months), implementations MAY stop producing events with the old algorithm.

During migration, the hash chain MAY contain events signed with different algorithms. Verifiers MUST support all algorithms that appear in the chain being verified.

4.4.1. Post-Quantum Readiness

Audit trails with multi-decade retention periods (up to 10 years at Gold level) face exposure to "harvest now, decrypt later" attacks. NIST has announced timelines for deprecating RSA and ECC algorithms (targeted deprecation by 2030, disallowance by 2035).

Implementations with Gold-level retention requirements SHOULD prepare for post-quantum transition by:

* Tracking ML-DSA (FIPS 204) readiness in their cryptographic libraries.

* Planning for hybrid classical+PQ signing approaches during the transition period, per the guidance in [RFC9794].

* Ensuring that signed Evidence Packs can be re-signed with post-quantum algorithms when available, while preserving the original classical signature chain for historical verification.

5. Common Event Structure

All VAP-conformant events MUST include the following fields:

```
{
  "vap_version": "1.3",
  "profile": {
    "id": "string (VCP|CAP|LAP|DVP|MAP|PAP)",
    "version": "semver string"
  },
  "header": {
    "event_id": "UUIDv7 (RFC 9562)",
    "chain_id": "UUIDv7",
    "prev_hash": "sha-256:<64 lowercase hex chars> | null",
    "timestamp": "RFC 3339 datetime with timezone",
    "event_type": "string (profile-specific)",
    "causal_link": {
      "target_event_id": "UUIDv7 | null",
      "link_type": "string (OUTCOME_OF|OVERRIDE_OF|HOLD_ON|
                            RECOVERY_OF|TIER_CHANGE_OF|null)"
    }
  },
  "provenance": {
    "actor": {
      "actor_id": "string",
      "actor_hash": "sha-256:<64 lowercase hex chars>",
      "role": "string"
    },
    "input": { },
    "context": { },
    "action": { },
    "outcome": { }
  },
  "accountability": {
    "operator_id": "string",
    "last_approval_by": "string",
    "approval_timestamp": "RFC 3339"
  },
  "domain_payload": { },
  "security": {
    "event_hash": "sha-256:<64 lowercase hex chars>",
    "hash_algo": "sha-256",
    "signature": "ed25519:base64url...",
    "sign_algo": "ed25519",
    "signer_id": "string"
  }
}
```

Event identifiers MUST use UUIDv7 ([RFC9562]) to ensure time-ordered sortability. JSON canonicalization MUST follow [RFC8785].

The profile.id field MUST be 1-4 uppercase ASCII characters matching a registered profile identifier (see Section 21.2). Note that profile identifiers use uppercase (e.g., "LAP") while algorithm identifiers use lowercase (e.g., "sha-256") per Table 2; this distinction is intentional and reflects their different registries.

Timestamps MUST conform to [RFC3339] (a profile of ISO 8601) and MUST include a timezone offset or "Z" for UTC. Implementations SHOULD use UTC for all timestamps.

The header.causal_link field provides a standardized location for referential integrity across all profiles. Outcome events MUST set target_event_id to the originating attempt event's identifier and link_type to "OUTCOME_OF". HUMAN_OVERRIDE events MUST set target_event_id to the target output event and link_type to "OVERRIDE_OF". Events with no causal link MUST set both fields to null.

Hash values MUST be encoded as lowercase hexadecimal strings, prefixed with the canonical algorithm identifier (from Table 2) followed by a colon (e.g., "sha-256:a1b2c3..."). The hexadecimal string MUST be the exact length corresponding to the hash output (64 characters for sha-256, 96 for sha-384, 128 for sha-512).

All text fields MUST be encoded as UTF-8 ([RFC3629]).

5.1. Numeric Value Encoding

Fields representing monetary amounts, cryptographic values, or high-precision measurements SHOULD be encoded as JSON strings rather than JSON numbers. This recommendation is motivated by:

* IEEE 754 double-precision floating-point, the only numeric type in JSON (per [RFC8259], Section 6), cannot exactly represent all decimal values. Financial and legal contexts require exact decimal representation.

* JSON parsers across programming languages exhibit inconsistent behavior for large integers (exceeding 2^53) and high-precision decimals, leading to silent data corruption.

* Canonicalization stability: [RFC8785] defines specific rules for numeric serialization, but string encoding avoids parser-dependent numeric reformatting entirely, ensuring consistent hash computation across implementations.

Fields where exact precision is not critical (e.g., event_count, token_count) MAY use JSON numbers. Implementations MUST document which fields use string encoding. Implementations that use JSON numbers for counters MUST ensure that any numeric-to-string conversion performed during canonicalization is deterministic and documented, to avoid signature verification ambiguity across languages and libraries.

6. Conformance Levels

VAP defines three conformance levels applicable to all domain profiles. Each level inherits all requirements of lower levels (Gold is a superset of Silver, which is a superset of Bronze).

6.1. Bronze Level

Target: SMEs, early adopters. Core capabilities:

* Event logging for all AI decision points (REQUIRED)
* Hash chain linking all events using a supported hash algorithm (REQUIRED)
* Digital signature on every event using a supported signature algorithm (REQUIRED)
* [RFC3339] timestamps with timezone (REQUIRED)
* UUIDv7 event identifiers (REQUIRED)
* Minimum 6-month retention (REQUIRED)
* Event structure validation against the Common Event Structure (Section 5) with at minimum all REQUIRED fields present and correctly typed (REQUIRED)

Note: Formal JSON Schema definitions for validation are provided in Appendix "Appendix B. Validation Requirements". Bronze implementations MUST validate the presence and type of all REQUIRED fields defined in Section 5.

6.2. Silver Level

Target: Enterprise, regulated industries. Additional requirements beyond Bronze:

* Daily external anchoring to a trusted timestamping service conforming to [RFC3161] (with [RFC5816] ESSCertIDv2 support) or an equivalent transparency log (REQUIRED)

* Completeness Invariant verification (REQUIRED)

* Evidence Pack generation capability (REQUIRED)

* Sensitive data hashing for privacy preservation (REQUIRED)

* Minimum 2-year retention (REQUIRED)

* Merkle tree construction per Section 7.3 (REQUIRED)

* Third-party verification endpoint conforming to Section 12 (REQUIRED)

6.3. Gold Level

Target: Highly regulated industries. Additional requirements beyond Silver:

* Hourly external anchoring (REQUIRED)

* HSM for signing key storage, [FIPS140-3] Level 2 or above (REQUIRED). Note: FIPS 140-2 validated modules MAY be used during the transition period ending September 2026, after which FIPS 140-3 is REQUIRED.

* Integration with a transparency log service such as IETF SCITT [IETF-SCITT] or equivalent (REQUIRED)

* Real-time audit API conforming to Section 12 (REQUIRED)

* Minimum 5-year retention (REQUIRED)

* 24-hour incident response and evidence preservation (REQUIRED)

* Geographic redundancy, minimum 2 regions (REQUIRED)

* Annual third-party audit (REQUIRED)

* Crypto-shredding support (REQUIRED)

7. External Anchoring

External anchoring proves that events existed at a specific point in time, preventing backdating, forward-dating, and log forking.

7.1. Anchoring Service Types

VAP defines an abstract anchoring interface that can be realized by multiple service types. The baseline anchoring service is [RFC3161] Time-Stamp Authority (TSA), with [RFC5816] support for ESSCertIDv2 (enabling SHA-256 certificate identification instead of SHA-1). Additional service types include transparency logs and public blockchains.

RFC 3161 TSA (Baseline) Traditional enterprise timestamping via X.509 PKI ([RFC3161]). This is the normative baseline. Trust model: CA trust hierarchy. Implementations MUST support [RFC5816] for SHA-256-based certificate identification.

Transparency Log (e.g., IETF SCITT) Append-only transparency logs providing public verifiability. IETF SCITT ([IETF-SCITT]) is one such service; implementations MAY use any transparency log providing equivalent append-only and inclusion-proof guarantees. Registration via SCRAPI ([IETF-SCRAPI]) is RECOMMENDED for SCITT integration. Trust model: public append-only log with cryptographic inclusion proofs.

Blockchain Bitcoin or Ethereum anchoring for maximum decentralization. Trust model: PoW/PoS consensus. This option is non-normative and provided for environments requiring decentralized trust.

Gold Level implementations MUST use at least one transparency log service (such as SCITT) or equivalent, in addition to or instead of RFC 3161 TSA. Implementations SHOULD use multiple independent anchoring services for critical deployments.

7.2. Anchor Record Format

```
{
  "anchor_id": "UUIDv7",
  "anchor_type": "RFC3161 | TRANSPARENCY_LOG | BLOCKCHAIN",
  "merkle_root": "sha-256:<64 lowercase hex chars>",
  "event_count": 1000,
  "first_event_id": "UUIDv7",
  "last_event_id": "UUIDv7",
  "first_event_timestamp": "RFC 3339",
  "last_event_timestamp": "RFC 3339",
  "anchor_timestamp": "RFC 3339",
  "anchor_proof": { },
  "service_endpoint": "https://tsa.example.com"
}
```

The anchor_proof field is an object whose structure depends on the anchor_type:

RFC3161 anchor_proof MUST contain: "tst_token" (Base64url-encoded RFC 3161 TimeStampToken per [RFC4648] Section 5), "hash_algo" (algorithm used in TSA request), and "tsa_cert_hash" (SHA-256 hash of the TSA certificate). Verification: parse the TimeStampToken, confirm the imprint matches the merkle_root, and validate the TSA signature chain.

TRANSPARENCY_LOG anchor_proof MUST contain: "inclusion_proof" (array of Base64url-encoded sibling hashes from leaf to root), "tree_size" (integer, total leaves at time of inclusion), "leaf_index" (integer, position of this entry), and "log_id" (identifier of the transparency log). Verification follows the Merkle audit path algorithm defined in [RFC9162] Section 2.1.3.

BLOCKCHAIN anchor_proof MUST contain: "tx_id" (transaction identifier as hex string), "block_height" (integer), "chain" (e.g., "bitcoin", "ethereum"), and "confirmations" (integer, number of confirmations at recording time). This anchor type is non-normative.

7.3. Merkle Tree Construction

Events MUST be batched into a binary Merkle hash tree for efficient anchoring and selective disclosure. The tree construction follows [RFC9162] (Certificate Transparency Version 2.0) Section 2, which obsoletes [RFC6962]:

```
MTH({})       = SHA-256()    (empty hash for zero inputs)
MTH({d(0)})   = SHA-256(0x00 || d(0))
MTH(D[n])     = SHA-256(0x01 || MTH(D[0:k]) || MTH(D[k:n]))

where:
  D[n] is the list of n event hashes
  k is the largest power of 2 less than n
  0x00 is the leaf node prefix
  0x01 is the interior node prefix
  || denotes concatenation
```

This construction uses domain separation prefixes (0x00 for leaves, 0x01 for interior nodes) to prevent second-preimage attacks, and handles non-power-of-two leaf counts without duplicating leaves, consistent with [RFC9162].

The resulting Merkle root is submitted to the external anchoring service. Merkle inclusion proofs enable selective disclosure: a verifier can confirm that a specific event is included in an anchored batch without accessing other events in the batch.

8. Completeness Invariant

The Completeness Invariant is a mathematical guarantee that every "attempt" event has exactly one corresponding "outcome" event. This prevents selective logging -- the omission of inconvenient records.

General form:

```
  For each pipeline P:
    Count(P_ATTEMPT) = Count(P_SUCCESS)
                     + Count(P_DENY)
                     + Count(P_ERROR)
```

The invariant enforces three properties:

Completeness Every ATTEMPT has an outcome. Violation indicates missing events.

Uniqueness Each ATTEMPT has exactly one outcome. Violation indicates duplicate records.

Referential Integrity Every outcome contains a causal link (via header.causal_link.target_event_id) to its originating ATTEMPT. Violation indicates orphan events.

Domain profiles MUST specify which event types constitute attempts and outcomes for the invariant. Each outcome event MUST set header.causal_link.target_event_id to the originating attempt event's event_id and header.causal_link.link_type to "OUTCOME_OF".

Verification SHOULD account for a configurable grace period for in-flight operations. The grace period MUST NOT exceed 300 seconds (RECOMMENDED default: 60 seconds). If an ATTEMPT event has no corresponding outcome event after the grace period has elapsed, a verification implementation SHOULD treat this as a completeness violation. Implementations MAY emit a synthetic TIMEOUT_ERROR outcome event when the grace period expires, to maintain the invariant.

9. Evidence Pack Specification

An Evidence Pack is a self-contained, signed package of provenance events suitable for regulatory submission and third-party audit.

9.1. Pack Structure

An Evidence Pack MUST contain:

* manifest.json: Pack metadata and integrity information
* events/: Event batches (max 10,000 events per file)
* anchors/: External anchor records
* merkle/: Merkle tree structure and selective disclosure proofs
* keys/: Public keys for signature verification
* signatures/: Pack-level signature

The Evidence Pack SHOULD be packaged as a ZIP archive with the media type application/vap-evidence-pack+zip (see Section 21.1). The manifest MUST use the media type application/vap-manifest+json.

9.2. Pack Manifest

The manifest MUST include the following fields:

pack_id (REQUIRED) UUIDv7 uniquely identifying this Evidence Pack.

vap_version (REQUIRED) VAP framework version (e.g., "1.3").

profile (REQUIRED) Object containing profile id and version.

conformance_level (REQUIRED) "Bronze", "Silver", or "Gold".

generated_at (REQUIRED) [RFC3339] timestamp of pack generation.

time_range (REQUIRED) Object with start and end [RFC3339] timestamps.

statistics (REQUIRED) Object containing total_events and events_by_type breakdown.

completeness_verification (REQUIRED for Silver+) Object containing invariant_type, invariant_valid boolean, grace_period_seconds, and per-pipeline results.

integrity (REQUIRED) Object containing checksums (SHA-256 per file), merkle_root, and pack_hash.

external_anchors (REQUIRED for Silver+) Array of anchor records

referencing this pack's time range.

retention_status (REQUIRED for Silver+ in LAP) Object containing events_at_tier1, events_at_tier2, events_at_tier3 counts, and active_legal_holds count and identifiers. See Section 18.

enforcement_metrics (REQUIRED for Silver+ in LAP) Object containing enforcement_level, warnings_issued, gates_blocked, gates_overridden, and rapid_approvals count and percentage. See Section 16.

The manifest MAY include additional profile-specific fields as defined by the domain profile specification.

10. Privacy-Preserving Verification

VAP enables verification of system integrity without disclosure of sensitive data. This is achieved through:

1. Hash-based attestation: Sensitive fields are stored as cryptographic hashes, enabling existence verification without content disclosure.

2. Selective disclosure via Merkle proofs: Individual events can be proven to exist within an Evidence Pack without revealing other events.

3. Per-tenant salting: Hash salts are unique per tenant to prevent cross-tenant correlation attacks.

This mechanism is particularly critical for LAP, where attorney-client privilege prevents disclosure of consultation content while still requiring verifiable audit trails.

10.1. Salt Management Requirements

Per-tenant salts MUST meet the following requirements:

* Minimum entropy: 256 bits (32 bytes), generated using a cryptographically secure random number generator.

* Uniqueness: Each tenant MUST have a distinct salt. Implementations MUST NOT share salts across tenants.

* Storage: Salts MUST be stored separately from the hashed data and protected with access controls equivalent to signing keys.

* Rotation: Salts SHOULD be rotated at least annually. Upon rotation, previously hashed values are NOT re-hashed; the provenance chain records the salt epoch in effect at each event's timestamp.

For fields with low input entropy (such as bar registration numbers, which occupy a bounded numeric space), plain SHA-256(salt || value) is vulnerable to dictionary attack if the salt is compromised. For such fields, implementations MUST use HMAC-SHA-256 with the tenant salt as key:

```
BarNumberHash = "sha-256:" || hex(HMAC-SHA-256(tenant_salt,
                                               bar_number_bytes))
```

In the event of salt compromise, the implementation MUST:

1. Generate a new salt immediately.

2. Record a SALT_ROTATION event in the provenance chain.

3. Assess and document the exposure window (events between last rotation and compromise discovery).

4. Notify affected tenants per applicable data breach notification requirements.

11. Retention Framework

Level	Events	Anchor Records	Evidence Packs	Keys
Bronze	6 months	N/A	On-demand	1 year after last use
Silver	2 years	5 years	2 years	3 years after last use
Gold	5 years	10 years	5 years	7 years after last use

Table 3: Retention Requirements by Conformance Level

Retention periods MUST be extended upon: regulatory investigation notification, legal hold orders, security or safety incidents, and third-party audit requests.

Domain profiles MAY specify extended retention periods beyond the VAP baseline where domain-specific regulations require longer retention (see Section 18 for LAP extensions).

11.1. Privacy Regulation Interaction

For privacy regulation compliance (e.g., [GDPR] "right to be forgotten"), implementations at Silver level and above SHOULD support crypto-shredding: encrypting personal data with per-user keys and deleting those keys to render the data cryptographically unrecoverable while preserving hash chain integrity.

VAP audit trails are designed to minimize tension with data protection regulations:

* Audit entries use hash-based references to personal data rather than storing personal data directly in the provenance chain. This reduces the personal data footprint within the immutable hash chain.

* Audit trails constitute a separate processing purpose with an independent legal basis. Under [GDPR] Article 6(1)(c) (legal obligation) or Article 6(1)(f) (legitimate interests in maintaining system integrity and accountability), provenance data may be retained independently of the subject’s consent.

* [GDPR] Article 17(3) provides exceptions to the right of erasure for legal claims (subparagraph (e)) and legal obligations (subparagraph (b)). Audit trail records documenting AI system compliance may fall within these exceptions, subject to jurisdiction-specific determination.

NOTE: The applicability of specific GDPR exceptions to VAP audit trails is a legal determination outside the scope of this specification. Implementations SHOULD consult legal counsel regarding data protection compliance in their deployment jurisdictions.

11.2. Content Retention Tiers

Implementations face a tension between privacy-preserving verification (which favors early deletion of sensitive content) and legal discovery obligations (which may require content disclosure). To address this, VAP defines three content retention tiers that domain profiles MAY adopt:

Tier 1 - Full Content Retention All event content (inputs, outputs,

documents) is retained in encrypted form alongside provenance hashes. Duration: configurable per tenant. Encryption MUST use AES-256-GCM or ChaCha20-Poly1305 with per-tenant keys.

Tier 2 - Recoverable Hash Retention Original content is deleted but a content recovery key is escrowed with a designated custodian. The escrowed key enables re-association of content from encrypted backups if a legal hold or disclosure order is triggered. Duration: from end of Tier 1 to the applicable retention period endpoint. See Section 11.3 for custodian requirements.

Tier 3 - Hash-Only Retention Only provenance hashes remain. Content is cryptographically unrecoverable (crypto-shredding complete). Duration: remainder of the retention period.

Transition from Tier 1 to Tier 2 MUST NOT occur while any Legal Hold is active for the affected events. Transition from Tier 2 to Tier 3 MUST NOT occur while any Legal Hold is active.

Implementations MUST log all tier transitions as RETENTION_TIER_CHANGE events in the provenance chain.

11.3. Escrow Custodian Requirements

Tier 2 escrow custodians MUST satisfy the following:

* The custodian MUST be organizationally independent from the VAP implementation operator (i.e., the entity that generates and signs provenance events).

* Key escrow operations (deposit, retrieval, destruction) MUST each generate a provenance event in the custodian’s own audit log.

* Escrow keys MUST be encrypted at rest using the custodian’s own key management infrastructure.

* Key granularity: implementations SHOULD use per-case or per-tenant escrow keys rather than a single global key, to limit the blast radius of any single key compromise.

* Recovery execution MUST require dual authorization: one from the requesting party (e.g., the tenant attorney) and one from the custodian.

A CONTENT_RECOVERY_EXECUTED event MUST be logged upon successful recovery, including: hold_id, recovered_event_range, recovered_by (actor hash), custodian_id, and court_order_reference_hash (if applicable).

11.4. Legal Hold Protocol

A Legal Hold freezes the current retention tier for all events within scope, preventing content deletion or tier transition.

Legal Hold triggers:

1. Court Order: Judicial order for evidence preservation.

2. Regulatory Investigation: Government regulatory action.

3. Litigation Hold: Reasonably anticipated litigation.

4. Professional Body Inquiry: Investigation by a professional regulatory body (e.g., bar association).

5. Disciplinary Proceedings: Professional disciplinary matters.

When a Legal Hold is activated:

1. All events within scope MUST be frozen at their current retention tier (Tier 1 or Tier 2).

2. A LEGAL_HOLD_ACTIVATED event MUST be recorded in the provenance chain, including: hold_id (UUIDv7), hold_trigger (enumerated trigger type), scope (event time range or case identifiers), activated_by (actor identity hash), and activation_timestamp ([RFC3339]).

3. Tier transitions for in-scope events MUST be blocked until a corresponding LEGAL_HOLD_RELEASED event is recorded.

4. The Legal Hold itself MUST be included in Evidence Packs generated during the hold period.

Legal Hold activation MUST be available at Bronze level. Automated Legal Hold detection (e.g., triggered by court filing notifications or regulatory inquiry receipt) is RECOMMENDED at Gold level.

11.5. Judicial Disclosure Response

When a court or regulatory body orders full content disclosure for specific events, the response depends on the current retention tier:

Events at Tier 1 Content is available in encrypted form and can be disclosed subject to appropriate access controls and professional review.

Events at Tier 2 Content recovery key is retrieved from escrow per Section 11.3. Content is reconstructed from encrypted backups. A CONTENT_RECOVERY_EXECUTED event MUST be logged in the provenance chain.

Events at Tier 3 Content is cryptographically unrecoverable. The implementation MUST provide:

1. Hash chain integrity proof demonstrating unbroken provenance.

2. Tier transition log showing when and why content was deleted.

3. Certification that no Legal Hold was active at the time of deletion.

4. Statistical metadata (token counts, timestamps, event types) that remains available.

This three-tier approach enables implementations to demonstrate to judicial authorities that content deletion followed a documented, auditable process rather than constituting potential evidence spoliation.

NOTE: The adequacy of hash-only evidence for specific judicial proceedings is a jurisdiction-specific legal determination outside the scope of this specification. This framework provides the maximum available technical evidence in each retention tier.

12. Third-Party Verification Protocol

Verification is available at three access levels:

Public Access to Merkle roots only. Verifies anchor existence.

Auditor Access to Evidence Packs. Full chain and completeness verification.

Regulator Real-time API access (Gold level). Live monitoring capability.

Verification steps:

1. Anchor Verification: Confirm Merkle root in external timestamping service or transparency log.

2. Chain Verification: Validate hash chain integrity from genesis to latest event.

3. Signature Verification: Authenticate all events with public keys.

4. Completeness Verification: Check invariant for the time period.

5. Selective Query (optional): Verify specific events via Merkle proofs.

12.1. Minimum Verification API

Silver and Gold level implementations that expose a third-party verification endpoint MUST support at minimum the following HTTP endpoints. All responses MUST use Content-Type application/json and MUST include appropriate authentication (e.g., Bearer token, mutual TLS).

GET /vap/v1/anchors?from={RFC3339}&to={RFC3339} Returns anchor records for the specified time range. Response: JSON array of Anchor Record objects per Section 7.2. REQUIRED for Silver+.

GET /vap/v1/chain/verify?from={RFC3339}&to={RFC3339} Performs and returns chain integrity verification results for the specified time range. Response: JSON object with fields: chain_valid (boolean), events_verified (integer), first_event_id, last_event_id, errors (array of {event_id, error_type, detail}). REQUIRED for Silver+.

GET /vap/v1/completeness?from={RFC3339}&to={RFC3339} Returns completeness invariant verification for the time range. Response: JSON object with fields: invariant_valid (boolean), grace_period_seconds (integer), pipelines (array of {pipeline_id, attempts, outcomes, valid}). REQUIRED for Silver+.

GET /vap/v1/events/{event_id}/proof Returns Merkle inclusion proof for a specific event. Response: JSON object with fields: event_id, anchor_id, merkle_root, inclusion_proof (array of Base64url sibling hashes), leaf_index, tree_size. REQUIRED for Silver+.

GET /vap/v1/events/stream (WebSocket or SSE) Real-time event stream for live monitoring. REQUIRED for Gold only.

Error responses MUST use standard HTTP status codes (400 for malformed requests, 401/403 for authentication/authorization failures, 404 for unknown resources, 500 for server errors) with a JSON body containing "error" (string code) and "detail" (human-readable description).

13. Legal AI Profile (LAP) Overview

The Legal AI Profile (LAP) is a VAP domain profile for judicial AI and LegalTech systems. LAP addresses unique challenges in the legal domain:

Unauthorized Practice of Law Risk Proving that AI does not independently practice law, through HUMAN_OVERRIDE events documenting attorney oversight.

Hallucination Recording fact-check provenance through LEGAL_FACTCHECK events with citation chain verification.

Selective Logging Preventing omission of inconvenient AI outputs through three-pipeline Completeness Invariant.

Attorney-Client Privilege Maintaining confidentiality through privacy-preserving fields (prompt hashes instead of raw content) and tiered content retention with legal hold support.

Accountability Ambiguity Recording "who, when, and on what basis" through the Accountability Layer and graduated override enforcement.

13.1. Profile Registration

Field	Value
Profile ID	LAP
Full Name	Legal AI Profile
Domain	Legal AI / LegalTech
Regulatory Scope	Attorney regulation, AI governance (informative)
Time Precision	Second
Profile Version	0.4.0

Table 4: LAP Profile Registration

14. LAP Event Type Taxonomy

LAP defines three functional pipelines, one cross-cutting control event type, and administrative event types for retention and enforcement management:

14.1. Pipeline 1: Legal Query

AI-powered legal consultation:

* LEGAL_QUERY_ATTEMPT: Question submission to AI
* LEGAL_QUERY_RESPONSE: AI response generated successfully
* LEGAL_QUERY_DENY: Response refused (content filter, unauthorized role)
* LEGAL_QUERY_ERROR: System error (API failure, timeout)

14.2. Pipeline 2: Document Generation

AI-assisted legal document drafting:

* LEGAL_DOC_ATTEMPT: Document generation request
* LEGAL_DOC_RESPONSE: Document generated successfully
* LEGAL_DOC_DENY: Generation refused (insufficient consent, unauthorized)
* LEGAL_DOC_ERROR: System error (API failure, parse error)

14.3. Pipeline 3: Fact Check

AI-powered legal fact verification:

* LEGAL_FACTCHECK_ATTEMPT: Fact-check request
* LEGAL_FACTCHECK_RESPONSE: Fact-check completed
* LEGAL_FACTCHECK_DENY: Fact-check refused (OPTIONAL)
* LEGAL_FACTCHECK_ERROR: System error

Implementations MAY define LEGAL_FACTCHECK_DENY for cases where a fact-check request is refused due to rate limiting, insufficient permissions, or consent constraints. The deny_reason field SHOULD distinguish these from system errors.

If an implementation does not support LEGAL_FACTCHECK_DENY, refusal conditions MUST be recorded as LEGAL_FACTCHECK_ERROR with a deny_equivalent indicator set to true in the error detail, ensuring the Completeness Invariant is maintained.

14.4. Cross-Cutting: Human Override

HUMAN_OVERRIDE events record an attorney's review of any AI output:

* APPROVE: Attorney confirms AI output without modification

* MODIFY: Attorney edits AI output (modification hash recorded)

* REJECT: Attorney rejects AI output entirely

HUMAN_OVERRIDE events MUST set header.causal_link.target_event_id to the target outcome event's identifier and header.causal_link.link_type to "OVERRIDE_OF". The event MUST also include the attorney's identity (bar number hash), override type, and optional modification details in the domain_payload.

14.5. Retention and Enforcement Events

LAP defines additional event types for retention management and override enforcement:

RETENTION_TIER_CHANGE Records transitions between content retention tiers. Fields: previous_tier, new_tier, affected_event_range, reason, authorized_by.

LEGAL_HOLD_ACTIVATED Records activation of a legal hold. Fields: hold_id, hold_trigger, scope, activated_by.

LEGAL_HOLD_RELEASED Records release of a legal hold. Fields: hold_id, released_by, release_reason.

CONTENT_RECOVERY_EXECUTED Records recovery of content from Tier 2 escrow. Fields: hold_id, recovered_event_range, recovered_by, custodian_id, court_order_reference_hash.

REVIEW_WARNING_ACKNOWLEDGED Records that an attorney acknowledged an unreviewed-output warning before proceeding with export. Fields: target_event_id, warning_type, acknowledged_by.

REVIEW_GATE_BLOCKED Records that an export attempt was blocked due to missing HUMAN_OVERRIDE. Fields: target_event_id, document_type, blocked_action.

REVIEW_GATE_OVERRIDE Records that an attorney bypassed a review gate with explicit justification. Fields: target_event_id, override_reason, overridden_by.

SALT_ROTATION Records rotation of tenant privacy salt. Fields: previous_salt_epoch, new_salt_epoch, rotated_by, reason.

These events are NOT part of the three-pipeline Completeness Invariant (they are control and administrative events, similar to HUMAN_OVERRIDE). However, they MUST be included in the hash chain and signed.

15. LAP Completeness Invariant

LAP applies the Completeness Invariant independently to all three pipelines:

For each pipeline P in {QUERY, DOC, FACTCHECK}:

```
Count(LEGAL_{P}_ATTEMPT)
= Count(LEGAL_{P}_RESPONSE)
+ Count(LEGAL_{P}_DENY)    [if supported]
+ Count(LEGAL_{P}_ERROR)
```

Expanded:

```
LEGAL_QUERY_ATTEMPT = LEGAL_QUERY_RESPONSE
                    + LEGAL_QUERY_DENY
                    + LEGAL_QUERY_ERROR

LEGAL_DOC_ATTEMPT   = LEGAL_DOC_RESPONSE
                    + LEGAL_DOC_DENY
                    + LEGAL_DOC_ERROR

LEGAL_FACTCHECK_ATTEMPT = LEGAL_FACTCHECK_RESPONSE
                        + LEGAL_FACTCHECK_DENY  [if supported]
                        + LEGAL_FACTCHECK_ERROR
```

For implementations that do not support LEGAL_FACTCHECK_DENY, the invariant simplifies to ATTEMPT = RESPONSE + ERROR for Pipeline 3. Refusal conditions recorded as ERROR with deny_equivalent MUST be counted toward the invariant.

Each outcome event MUST set header.causal_link.target_event_id to the originating attempt event's event_id and header.causal_link.link_type to "OUTCOME_OF", ensuring referential integrity can be verified independently of event ordering.

16. Override Coverage and Enforcement

16.1. Override Coverage Metric

HUMAN_OVERRIDE events are outside the Completeness Invariant but LAP defines Override Coverage as a critical operational metric:

```
Override Coverage =
  Count(distinct target_event_id with at least one
        HUMAN_OVERRIDE where link_type = "OVERRIDE_OF") /
  Count(LEGAL_*_RESPONSE)
```

This metric quantifies the degree to which human professionals review AI outputs. In jurisdictions where regulations require that a licensed professional personally scrutinize AI-generated work products, this metric provides measurable evidence of compliance.

Design notes on the denominator:

* The numerator counts distinct target events (not raw HUMAN_OVERRIDE count) to prevent a single output reviewed multiple times from inflating coverage above 100%.

* DENY events are excluded from the denominator because a denial represents the AI system refusing to produce output. There is no AI-generated work product for a professional to review. If a jurisdiction requires that denial decisions themselves be reviewed, the implementation SHOULD track this as a separate "Denial Review Coverage" metric.

* ERROR events are excluded from the denominator because they do not produce an output suitable for professional approval or rejection. Completeness of error handling is evaluated separately via the per-pipeline invariant.

Coverage	Assessment	Operational Implication
100%	Ideal	Full professional oversight of all AI outputs
70-99%	Good	Majority reviewed; low-risk outputs may be excluded
30-69%	Warning	Insufficient review; operational improvement recommended
<30%	Critical	Professional oversight requirements likely unmet

Table 5: Override Coverage Assessment

16.2. Enforcement Levels

Override Coverage tracking alone provides post-hoc accountability but does not prevent the use of unreviewed AI outputs in professional practice. LAP defines four enforcement levels to address this structural limitation:

Level 0 - Metric Only Override Coverage is computed and reported. No enforcement action is taken. The system relies on the professional's ethical obligations.

Level 1 - Warn When a user attempts to export, copy, or transmit an AI-generated output that has no corresponding HUMAN_OVERRIDE event, the system MUST display a prominent warning indicating that the output has not been professionally reviewed.

The warning MUST: (a) be displayed in-context at the point of export or copy action; (b) require explicit acknowledgment before proceeding; and (c) record a REVIEW_WARNING_ACKNOWLEDGED event in the provenance chain.

Level 2 - Gate For designated high-risk document types, the system MUST require a HUMAN_OVERRIDE event before permitting export or transmission.

The gate MUST: (a) block export, copy, and transmit actions until a HUMAN_OVERRIDE (APPROVE or MODIFY) event exists for the target output; (b) record a REVIEW_GATE_BLOCKED event when an export attempt is blocked; and (c) allow the gate to be bypassed ONLY by a user with "attorney" role AND explicit override, recorded as a REVIEW_GATE_OVERRIDE event with a mandatory reason field.

Document types subject to gating SHOULD be configurable per tenant. The default gated types are: court filings, settlement and mediation documents, client-facing legal opinions, and contracts and agreements.

Level 3 - Strict ALL AI-generated outputs require HUMAN_OVERRIDE before any export action. No bypass mechanism. This level is intended for maximum-compliance environments but is not mandated by this specification due to potential impact on workflow efficiency.

16.3. Enforcement Level Mapping

Enforcement	Bronze	Silver	Gold
Level 0 (Metric Only)	Default	Minimum	N/A
Level 1 (Warn)	OPTIONAL	REQUIRED	REQUIRED
Level 2 (Gate)	N/A	RECOMMENDED	RECOMMENDED
Level 3 (Strict)	N/A	OPTIONAL	OPTIONAL

Table 6: Enforcement Level by Conformance Level

16.4. Override Latency Threshold

To distinguish genuine professional review from perfunctory approval, LAP defines an Override Latency metric:

```
Override Latency = HUMAN_OVERRIDE.timestamp
                 - target_output_event.timestamp
```

Implementations at Silver level and above SHOULD flag HUMAN_OVERRIDE events with Override Latency below a configurable threshold (RECOMMENDED default: 10 seconds) as "rapid approval" in the provenance chain.

This does NOT block the override but records a RAPID_APPROVAL_FLAG in the event metadata, which: (a) is visible in Evidence Packs and audit reports; (b) may indicate insufficient review depth; and (c) can trigger alerts at Gold level when rapid approvals exceed a configurable percentage (RECOMMENDED: 20%).

16.5. Structural Limitation Acknowledgment

This specification acknowledges that no technical mechanism can fully guarantee the quality or depth of human professional review. A licensed attorney may approve an AI output after genuine review or after cursory inspection; the system cannot distinguish between these without introducing unacceptable surveillance of professional judgment.

The enforcement framework therefore aims to: (a) create friction against entirely unreviewed AI output usage; (b) provide auditable evidence of review or lack thereof; (c) enable risk-proportionate controls (stronger for high-risk document types); and (d) preserve professional autonomy while recording accountability metadata.

The combination of enforcement levels, latency monitoring, and comprehensive provenance logging shifts the accountability model from "trust alone" to "trust with verification infrastructure," acknowledging that while absolute prevention is impossible, the cost and detectability of non-compliance can be substantially increased.

17. LAP Privacy-Preserving Fields

Legal AI handles extremely sensitive data protected by professional privilege. LAP extends VAP’s privacy-preserving verification with the following hashed fields:

Original Data	Hash Field	Hash Method	Sensitive Content
User query text	PromptHash	SHA-256(salt || content)	Legal consultation content (privileged)
AI response text	ResponseHash	SHA-256(salt || content)	AI-generated legal advice
Document output	OutputHash	SHA-256(salt || content)	Generated legal documents
Case number	CaseNumberHash	HMAC-SHA-256(salt, value)	Case identifier (high specificity)
Bar number	BarNumberHash	HMAC-SHA-256(salt, value)	Professional registration number
Party names	PartyHash	HMAC-SHA-256(salt, value)	Personal information of parties
Modification detail	ModificationHash	SHA-256(salt || content)	Professional's corrections
Factcheck content	TargetContentHash	SHA-256(salt || content)	Content under verification

Table 7: LAP Privacy-Preserving Fields

Fields with low input entropy (CaseNumberHash, BarNumberHash, PartyHash) MUST use HMAC-SHA-256 with the tenant salt as key, per Section 10.1, to resist dictionary attacks. Fields with high input entropy (content hashes) MAY use simple salted SHA-256.

Hash computation uses per-tenant salts as specified in Section 10.1. Third-party verifiers can confirm event existence and chain integrity without accessing privileged content.

The availability of original content corresponding to these hashes depends on the current Content Retention Tier (Section 11.2). At Tier 1, original content is available in encrypted form. At Tier 2, it is recoverable via escrowed keys. At Tier 3, only hashes remain.

18. LAP Conformance Level Mapping

Requirement	Bronze	Silver	Gold
Hash Chain	Yes	Yes	Yes
Digital Signature	Yes	Yes	Yes
External Anchoring	No	Daily	Hourly
Completeness Invariant	No	3 Pipelines	3 Pipelines
Override Coverage Tracking	No	Yes	Yes (with alerts)
Override Enforcement Level	Level 0	Level 1 (REQUIRED)	Level 1 (REQUIRED)
Evidence Pack	No	Yes	Yes
Privacy Hashing	No	Yes	Yes
Content Retention Tiers	Tier 3 only	Tier 2 + Tier 3	All 3 Tiers
Legal Hold Protocol	Yes (manual)	Yes (manual)	Yes (automated detection)
Content Recovery Escrow	No	RECOMMENDED	REQUIRED
HSM	No	No	Yes ([FIPS140-3])
Retention	6 months	3 years	10 years
Real-time Audit API	No	No	Yes

Table 8: LAP Conformance Requirements

19. LAP Regulatory Alignment (Informative)

19.1. Attorney Professional Regulation

In many jurisdictions, only licensed attorneys may provide legal advice. AI systems that generate legal analysis or draft legal documents operate in a regulatory space where the boundary between "legal information" and "legal advice" determines whether unauthorized practice of law has occurred.

LAP provenance supports regulatory compliance through:

* Actor.role and BarNumberHash: supports verification that the user is a licensed attorney.

* HUMAN_OVERRIDE (APPROVE/MODIFY): supports demonstrating attorney scrutiny.

* ModificationHash: supports evidence of attorney modifications.

* Enforcement Level 1/2: provides system-level friction against use of unreviewed outputs.

* Override Latency monitoring: flags potentially insufficient review.

In Japan, the [JAPAN-ATTORNEY-ACT] Article 72 prohibits unauthorized practice of law. The [MOJ-GUIDELINE] (August 2023) clarifies that AI contract review services may constitute unauthorized practice when crossing into dispute-related legal advice. The [JFBA-AI-GUIDANCE] (September 2025) requires that responsibility for legal advice remains with the attorney and that AI output cannot reach clients without attorney review. LAP’s Human Override Coverage metric provides verifiable evidence that these requirements are met.

Japan’s AI Promotion Act [JAPAN-AI-ACT] (enacted May 2025, effective September 2025) establishes transparency as a statutory principle. While no specific audit trail requirements are imposed, the Act’s agile governance model relies on private-sector technical standards for implementation, creating potential demand for frameworks like VAP.

19.2. High-Risk AI System Governance

Legal AI systems may be classified as high-risk under AI governance frameworks such as the [EU-AI-ACT], particularly under Annex III Section 8(a) which explicitly covers AI systems intended to "assist a judicial authority in researching and interpreting facts and the law."

LAP Silver level and above provides audit trail capabilities that can help satisfy the following [EU-AI-ACT] requirements:

* Article 12 (Automatic event logging): Hash chain and event logging provide automatic recording over the system lifetime.

* Article 14 (Human oversight): HUMAN_OVERRIDE events document human ability to override and reverse AI outputs.

* Article 18 (10-year documentation retention): Gold level supports 10-year retention.

* Article 19 (Minimum 6-month log retention): All conformance levels meet this minimum.

* Tiered content retention with legal hold supports discovery obligations under Article 12 and national procedural law.

NOTE: Enforcement timelines for Annex III high-risk AI obligations are subject to revision. The core technical requirements under Articles 12, 14, 18, and 19 remain stable regardless of enforcement timing. The degree to which LAP capabilities satisfy specific regulatory requirements should be evaluated on a per-jurisdiction basis.

20. Security Considerations

This section describes the threat model and security considerations for VAP-LAP implementations, following the guidance in [RFC3552].

20.1. Threat Model

VAP assumes the following attacker capabilities and trust assumptions:

* Attacker capabilities: An adversary may have read access to the event store, network access to observe event traffic, and in some scenarios write access to event storage (insider threat). The adversary’s goal is to forge, suppress, reorder, or selectively disclose provenance events.

* Trust assumptions: The signing key holder is trusted to sign only genuine events. The external anchoring service is trusted to provide accurate timestamps and append-only guarantees. Verifiers are trusted to correctly execute the verification algorithm.

20.2. Hash Chain Manipulation

An adversary with write access could attempt to modify past events, insert fabricated events, or delete events. The hash chain provides integrity: any modification to a past event changes its EventHash, breaking the chain at that point. External anchoring at Silver level and above makes post-hoc modification detectable by comparing the stored Merkle root against the independently timestamped root. Implementations SHOULD use Merkle consistency proofs (per [RFC9162] Section 2.1.4) to detect log forking.

20.3. Timestamp Manipulation

Timestamp rollback or clock skew can cause false completeness verification failures and undermine event ordering guarantees. Implementations SHOULD use monotonic time sources and SHOULD cross-validate local timestamps against external anchoring timestamps. External anchoring at Silver level and above provides an independent time reference. Implementations MUST reject events with timestamps that differ from the external anchor timestamp by more than a configurable bound (RECOMMENDED: 300 seconds for batch anchoring, 60 seconds for real-time anchoring).

20.4. Insider Threats

An authorized operator could forge events, suppress events before they are anchored, or create fabricated ERROR events to satisfy the Completeness Invariant while hiding actual outcomes. Mitigations include:

* External anchoring makes suppression detectable after anchoring.

* Gold level implementations SHOULD require multi-party signatures for administrative events (Legal Hold, tier changes).

* Separation of duties: the event signer and the external anchor submitter SHOULD be different principals where operationally feasible.

* Completeness Invariant cross-validation against application-layer logs (e.g., HTTP request logs) can detect fabricated events at audit time.

20.5. Key Compromise

Compromise of signing keys allows event forgery. Bronze implementations SHOULD rotate keys annually. Silver MUST rotate semi-annually. Gold MUST use HSM storage ([FIPS140-3]) and quarterly rotation.

Upon key compromise detection, the implementation MUST:

1. Immediately revoke the compromised key.

2. Generate a new signing key pair.

3. Record a KEY_ROTATION event signed by the new key, referencing the compromised key identifier.

4. Re-anchor all events signed with the compromised key using a fresh Merkle tree signed under the new key.

5. Notify all relying parties of the compromise and the validity window of the affected key.

During key rotation (including non-emergency rotation), the hash chain continues unbroken. The new key signs a KEY_ROTATION event that includes the old key's public key hash, establishing chain continuity. Verifiers MUST accept events signed by either key during the overlap period specified in the KEY_ROTATION event.

20.6. Privacy Leakage

Per-tenant salting prevents cross-tenant hash correlation. Implementations MUST NOT share salts across tenants. Event metadata (timestamps, event types, counts) may leak statistical information even when content is hashed; this is an inherent limitation of any audit trail system. Selective disclosure via Merkle proofs (Section 7.3) enables verifiers to inspect specific events without accessing the full event store, reducing exposure.

For privacy analysis guidance, see [RFC6973].

20.7. Availability Attacks

Denial-of-service attacks against the logging infrastructure could prevent event recording, violating completeness. Gold level implementations MUST have geographic redundancy. Implementations SHOULD buffer events locally when the central event store is unavailable, and MUST reconcile buffered events upon reconnection.

20.8. Denial of Provenance

A signer could repudiate previously signed events. External anchoring provides non-repudiation: the independently timestamped Merkle root proves the event existed at the anchor time. Combined with the signer's public key bound to their identity, this provides evidence against repudiation.

20.9. Selective Disclosure Attacks

An adversary could present only favorable events to a verifier, hiding unfavorable records. The Completeness Invariant detects such omissions: any missing outcome event will cause the invariant to fail for the affected pipeline. External anchoring of Merkle roots makes it infeasible to present a consistent subset of events that satisfies the invariant if any events have been omitted.

20.10. Replay Attacks

An adversary could replay valid events from one context into another. The chain_id field binds events to a specific chain, and the prev_hash field creates a strict ordering dependency that makes replayed events detectable (they will break the hash chain at the insertion point). UUIDv7 event identifiers provide globally unique, time-ordered identification that further constrains replay.

20.11. Content Retention and Discovery Risk

The tension between privacy-preserving hash-only retention and judicial discovery obligations creates a risk that hash-only evidence may be deemed insufficient by courts. The Tiered Retention model (Section 11.2) mitigates this by maintaining recoverable content during periods when discovery is most likely, while the Legal Hold Protocol (Section 11.4) prevents premature content destruction when litigation is anticipated.

20.12. Override Enforcement Circumvention

Enforcement Levels 1 and 2 (Section 16.2) can be circumvented by users who copy AI output through means outside the system's control (e.g., screen capture, manual transcription). Technical enforcement addresses the common case of system-mediated export but cannot prevent all forms of circumvention. The provenance chain nonetheless records the absence of HUMAN_OVERRIDE events, providing post-hoc accountability.

20.13. Rapid Approval Gaming

Users aware of the Override Latency threshold (Section 16.4) might delay approval clicks to avoid the RAPID_APPROVAL_FLAG. This is a known limitation. Statistical analysis of approval latency distributions across users and document types can detect such gaming patterns in audit reviews.

20.14. Legal Hold Integrity

Legal Hold activation and release events must be protected from unauthorized modification. Gold level implementations SHOULD require multi-party authorization for Legal Hold release.

21. IANA Considerations

21.1. Media Type Registration

IANA is requested to register the following media types in the "Media Types" registry:

21.1.1. application/vap-evidence-pack+zip

Type name: application

Subtype name: vap-evidence-pack+zip

Required parameters: None

Optional parameters: profile (string): VAP profile identifier (e.g., "LAP", "VCP")

Encoding considerations: binary (ZIP archive)

Security considerations: Evidence Packs contain signed provenance data. Implementations MUST verify the pack-level signature before processing. The pack may contain privacy-sensitive metadata; access control is the responsibility of the implementation. See Section 20 of this document.

Interoperability considerations: The internal structure follows Section 9.1. Implementations MUST support the manifest.json format defined in Section 9.2.

Published specification: This document

Applications that use this media type: AI audit trail systems, regulatory submission tools, third-party verification services

Fragment identifier considerations: N/A

Additional information: N/A

Contact: AILEX LLC, info@ailex.co.jp

Author: Shintaro Yamakawa

Change controller: IETF

21.1.2. application/vap-manifest+json

Type name: application

Subtype name: vap-manifest+json

Required parameters: None

Optional parameters: None

Encoding considerations: UTF-8 encoded JSON per [RFC8259]

Security considerations: The manifest is integrity-protected as part of the Evidence Pack signature. See Section 20.

Interoperability considerations: JSON format per Section 9.2.

Published specification: This document

Applications that use this media type: AI audit trail systems

Contact: AILEX LLC, info@ailex.co.jp

Author: Shintaro Yamakawa

Change controller: IETF

21.2. VAP Domain Profile Registry

IANA is requested to create a "VAP Domain Profile Identifiers" registry with the following initial entries. The registration policy is Specification Required per [RFC8126] Section 4.6.

Profile ID	Full Name	Domain	Reference
VCP	VeritasChain Protocol	Finance	This document
CAP	Content AI Profile	Content/ Creative AI	This document
LAP	Legal AI Profile	Legal AI / LegalTech	This document
DVP	Driving/Vehicle Profile	Automotive	Reserved
MAP	Medical AI Profile	Healthcare	Reserved
PAP	Public Admin Profile	Public Administration	Reserved

Table 9: VAP Domain Profile Identifiers

Each registration MUST include: Profile ID (1-4 uppercase ASCII characters), Full Name, Domain description, and a reference to the defining specification.

21.3. VAP Event Type Registry

IANA is requested to create a "VAP Event Types" registry. The registration policy is Specification Required per [RFC8126] Section 4.6. Initial entries include all event types defined in Section 14 of this document.

22. Relationship to Other Work

Several IETF efforts address aspects of AI accountability and provenance. This section clarifies how VAP relates to and complements these efforts.

IETF SCITT ([IETF-SCITT]) SCITT provides content-agnostic transparency infrastructure for digital supply chains. VAP Evidence Packs can be registered as SCITT Signed Statements, leveraging SCITT's append-only guarantees and inclusion proofs to strengthen VAP's external anchoring. VAP and SCITT are complementary: SCITT provides the transparency infrastructure, while VAP defines what goes into the payload (domain-specific provenance events with completeness guarantees).

IETF RATS / EAT (RFC 9711) The Remote Attestation Procedures (RATS) architecture and Entity Attestation Token (EAT) format address device and software integrity attestation. VAP could leverage EAT claims for attesting the integrity of AI model components within the provenance chain. The two frameworks operate at different layers: RATS/EAT attests what the system is, while VAP records what the system did.

IETF AIPREF Working Group AIPREF standardizes preferences about AI content usage (input-side concern). VAP addresses output-side accountability. The two are naturally complementary: VAP audit trails could prove that an AI system honored AIPREF preferences.

draft-reilly-sentinel-protocol The Sentinel Protocol defines evidence packages for AI lifecycle provenance covering datasets, training, and inference. VAP differentiates through: (1) domain-specific profiles (e.g., LAP) with regulatory mappings, (2) the Completeness Invariant preventing omission attacks, and (3) tiered conformance levels mapping to specific regulatory requirements. The two specifications could be complementary, with Sentinel addressing model lifecycle and VAP addressing operational decision provenance.

ISO/IEC Standards ISO/IEC 42001 (AI Management System), ISO/IEC 42005 (AI Impact Assessment), and the emerging ISO/IEC 24970 (AI System Logging) define architecture-agnostic requirements that VAP's technical mechanisms can satisfy. VAP conformance levels are designed to map to these frameworks' requirements.

23. References

23.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, DOI 10.17487/RFC8032, January 2017, <https://www.rfc-editor.org/info/rfc8032>.

[RFC8785] Rundgren, A., Jordan, B., and S. Erdtman, "JSON Canonicalization Scheme (JCS)", RFC 8785, DOI 10.17487/RFC8785, June 2020, <https://www.rfc-editor.org/info/rfc8785>.

[RFC9562] Davis, K., Peabody, B., and P. Leach, "Universally Unique IDentifiers (UUIDs)", RFC 9562, DOI 10.17487/RFC9562, May 2024, <https://www.rfc-editor.org/info/rfc9562>.

[RFC3161] Adams, C., Cain, P., Pinkas, D., and R. Zuccherato, "Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP)", RFC 3161, DOI 10.17487/RFC3161, August 2001, <https://www.rfc-editor.org/info/rfc3161>.

[RFC5816] Santesson, S. and N. Pope, "ESSCertIDv2 Update for RFC 3161", RFC 5816, DOI 10.17487/RFC5816, April 2010, <https://www.rfc-editor.org/info/rfc5816>.

[RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <https://www.rfc-editor.org/info/rfc3339>.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <https://www.rfc-editor.org/info/rfc4648>.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/info/rfc8259>.

[RFC9162] Laurie, B., Messeri, E., and R. Stradling, "Certificate Transparency Version 2.0", RFC 9162, DOI 10.17487/RFC9162, December 2021, <https://www.rfc-editor.org/info/rfc9162>.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November 2003, <https://www.rfc-editor.org/info/rfc3629>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

23.2. Informative References

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI 10.17487/RFC3552, July 2003, <https://www.rfc-editor.org/info/rfc3552>.

[RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, DOI 10.17487/RFC6973, July 2013, <https://www.rfc-editor.org/info/rfc6973>.

[RFC7696] Housley, R., "Guidelines for Cryptographic Algorithm Agility and Selecting Mandatory-to-Implement Algorithms", BCP 201, RFC 7696, DOI 10.17487/RFC7696, November 2015, <https://www.rfc-editor.org/info/rfc7696>.

[RFC9794] Driscoll, F., Parsons, M., and B. Hale, "Terminology for Post-Quantum Traditional Hybrid Schemes", RFC 9794, DOI 10.17487/RFC9794, June 2025, <https://www.rfc-editor.org/info/rfc9794>.

[RFC6962] Laurie, B., Langley, A., and E. Kasper, "Certificate Transparency", RFC 6962, DOI 10.17487/RFC6962, June 2013, <https://www.rfc-editor.org/info/rfc6962>.

[RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", BCP 205, RFC 7942, DOI 10.17487/RFC7942, July 2016, <https://www.rfc-editor.org/info/rfc7942>.

[RFC6979] Pornin, T., "Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA)", RFC 6979, DOI 10.17487/RFC6979, August 2013, <https://www.rfc-editor.org/rfc/rfc6979>.

[EU-AI-ACT] European Parliament and Council, "Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act)", Official Journal of the European Union, L 2024/1689, 2024.

[JAPAN-ATTORNEY-ACT] Government of Japan, "Attorney Act (Bengoshi-ho), Act No. 205 of 1949", 1949.

[MOJ-GUIDELINE]
Ministry of Justice, Japan, "Regarding the Relationship between AI-based Contract Document Support Services and Attorney Act Article 72", August 2023.

[JFBA-AI-GUIDANCE]
Japan Federation of Bar Associations, "Precautions Regarding the Use of Generative AI in Attorney Practice", September 2025.

[JAPAN-AI-ACT]
Government of Japan, "Act on Promotion of Development and Use of AI-related Technologies (AI Promotion Act)", May 2025.

[IETF-SCITT]
Birkholz, H., Delignat-Lavaud, A., Fournet, C., Deshpande, Y., and S. Lasker, "An Architecture for Trustworthy and Transparent Digital Supply Chains", Work in Progress, Internet-Draft, draft-ietf-scitt-architecture-22, 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-scitt-architecture-22>.

[IETF-SCRAPI]
Steele, O., "SCITT Reference API", Work in Progress, Internet-Draft, draft-ietf-scitt-scrapi-06, 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-scitt-scrapi-06>.

[GDPR]
European Parliament and Council, "Regulation (EU) 2016/679 - General Data Protection Regulation", 2016.

[FIPS203]
National Institute of Standards and Technology, "Module-Lattice-Based Key-Encapsulation Mechanism Standard", FIPS 203, August 2024.

[FIPS204]
National Institute of Standards and Technology, "Module-Lattice-Based Digital Signature Standard", FIPS 204, August 2024.

[FIPS140-3]
National Institute of Standards and Technology, "Security Requirements for Cryptographic Modules", FIPS 140-3, March 2019.

Appendix A. Profile Comparison

Aspect	VCP (Finance)	CAP (Content)	LAP (Legal)
Time Precision	Nanosecond	Second	Second
Key Invariant	SIG to ORD	GEN_ATTEMPT to GEN/DENY/ERROR	3 Pipeline Invariants
Unique Feature	Signal integrity	Safe Refusal (SRP)	Human Override Coverage + Enforcement
Regulatory Focus	Financial regulation	Content regulation	Attorney regulation + AI governance
Privacy Model	Trade data	Creative content	Professional privilege + Tiered Retention
Retention (Gold)	5 years	5 years	10 years

Table 10: Comparison of VAP Domain Profiles

Appendix B. Validation Requirements

Bronze level implementations MUST validate that all events conform to the Common Event Structure (Section 5). At minimum, validation MUST check:

* All REQUIRED top-level fields are present: vap_version, profile, header, provenance, accountability, security.

* header.event_id is a valid UUIDv7.

* header.timestamp conforms to [RFC3339].

* header.prev_hash is either null (genesis) or matches the pattern "{algo_id}:{hex_string}" where algo_id is a canonical algorithm identifier from Table 2.

* security.event_hash matches the recomputed hash over the Hash Input.

* security.signature is present and valid for the specified sign_algo.

* header.causal_link.target_event_id is either null or a valid UUIDv7.

Formal JSON Schema definitions will be published as a companion document or hosted at a stable URI referenced in a future revision of this specification.

Appendix C. Implementation Status

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC7942]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC7942], "this section, as well as the cited RFC, should be removed before the document is approved for publication as an RFC."

AILEX SaaS Platform Organization: AILEX LLC. Description: Production implementation of VAP-LAP for legal AI services including AI-powered consultation, document generation, and fact-checking. Maturity: Bronze-level conformance (hash chain integrity, digital signatures, event logging, UUIDv7 identifiers) with select Silver-level features implemented on an experimental basis: three-pipeline completeness invariant, privacy-preserving fields with per-tenant salting, and override coverage tracking. External anchoring and Evidence Pack generation (Silver REQUIRED) are not yet implemented. Licensing: Proprietary. URL: https://users.ailex.co.jp

Appendix D. Change Log

This section tracks changes between Internet-Draft revisions and will be removed before publication.

draft-ailex-vap-legal-ai-provenance-02

* Added individual author name (Shintaro Yamakawa) per IETF guidelines requiring personal author identification.

* Fixed cryptographic algorithm naming: Kyber-1024 renamed to ML-KEM-1024 per FIPS 203 standardization; separated into distinct KEM category from Encryption. Added ML-DSA-65 parameter set selection rationale.

* Resolved EventHash circular reference: defined explicit Hash Input excluding security.event_hash and security.signature from canonicalization input (Section 4.2).

* Resolved crypto agility contradiction: hash chain and signature formulas now parameterized by hash_algo/sign_algo fields instead of hardcoding SHA-256/Ed25519. Added Algorithm Migration procedure (Section 4.4) with post-quantum readiness subsection referencing FIPS 204 and RFC 9794.

* Updated Merkle tree construction to follow RFC 9162 (replacing obsoleted RFC 6962), with domain separation prefixes and correct handling of non-power-of-two leaf counts (Section 7.3).

* Fixed encoding specifications: timestamps now require RFC 3339; Base64 encoding now requires RFC 4648 Section 5 (URL-safe, no padding); hash values standardized as lowercase hex; RFC 8259 added to normative references (Section 5).

* Updated FIPS 140-2/3 to FIPS 140-3 as primary requirement with transition period note (Section 6.3).

* Added RFC 5816 (ESSCertIDv2) as normative reference for RFC 3161 TSA SHA-256 support (Section 7.1).

* Specified anchor_proof structure per anchor_type with type-specific fields and verification procedures (Section 7.2).

* Added Completeness Invariant grace period maximum of 300 seconds with TIMEOUT_ERROR mechanism (Section 8).

* Added standardized causal_link field in header for all event types, with link_type enumeration (Section 5).

* Fixed Override Coverage metric: numerator changed to count distinct target_event_id to prevent exceeding 100%. DENY removed from denominator with rationale documented (Section 16.1).

* Added salt management requirements: 256-bit minimum entropy, HMAC-SHA-256 for low-entropy fields, rotation and compromise response procedures (Section 10.1).

* Added minimum verification API with HTTP endpoints for Silver and Gold levels (Section 12.1).

* Added Bronze-level validation requirements specifying minimum field presence and type checking (Appendix B).

* Expanded Security Considerations with explicit threat model, 12 specific threat analyses per RFC 3552, key rotation procedures, and denial-of-provenance mitigation (Section 20).

* Expanded IANA Considerations with media type registration templates, VAP Domain Profile registry, and VAP Event Type registry per RFC 8126 (Section 21).

* Added GDPR interaction subsection addressing right-to- erasure vs. audit trail tension (Section 11.1).

* Added escrow custodian requirements for Tier 2 content recovery (Section 11.4).

* Added Relationship to Other Work section positioning VAP relative to SCITT, RATS/EAT, AIPREF, Sentinel Protocol, and ISO/IEC standards (Section 22).

* Added Implementation Status appendix per RFC 7942 (Appendix C).

* Added Japan AI Promotion Act (2025) to regulatory alignment references.

* Added SALT_ROTATION event type to LAP administrative events (Section 14.5).

* Removed EIP from profile.id enumeration (undefined profile).

* Normalized all algorithm identifiers to lowercase hyphenated ASCII strings with explicit canonical mapping table (Section 4.1). Resolved inconsistency between field values (e.g., "SHA-256") and wire prefixes (e.g., "sha256:") by unifying on a single form (e.g., "sha-256" / "sha-256:").

* Added UTF-8 encoding requirement for all text fields (Section 5, RFC 3629).

* Updated vap_version from 1.2 to 1.3.

* Updated LAP Profile Version from 0.3.0 to 0.4.0.

* Updated Abstract to reference SCITT.

draft-ailex-vap-legal-ai-provenance-01

* Added Tiered Content Retention model (Section 11.2).

* Added Legal Hold Protocol (Section 11.5) with five trigger types.

* Added Judicial Disclosure Response procedures (Section 11.6).

* Expanded Override Coverage with four-level Enforcement Framework.

* Added Override Latency Threshold with Rapid Approval Flag.

* Added Section 16.5 acknowledging structural limitations.

* Added seven new event types for retention and enforcement.

* Extended Evidence Pack manifest with retention_status and enforcement_metrics.

* Updated LAP Profile Version from 0.2.0 to 0.3.0.

draft-ailex-vap-legal-ai-provenance-00 Initial submission.

Acknowledgments

The VAP Framework and LAP Profile were developed with input from: the CAP v1.0 Safe Refusal Provenance (SRP) design experience, the VCP v1.1 operational feedback, regulatory engagement from legal practitioners, and open-source community contributions.

LAP v0.4 design draws from the AILEX SaaS reference implementation, the Ministry of Justice guideline on AI services and [JAPAN-ATTORNEY-ACT] Article 72 (August 2023), the [JFBA-AI-GUIDANCE] on generative AI in attorney practice (September 2025), and operational feedback from pilot law firms regarding judicial discovery and attorney oversight workflows.

The authors thank the reviewers who provided detailed technical feedback on the -01 draft, particularly regarding cryptographic algorithm naming accuracy, hash computation clarity, Merkle tree construction consistency, and IANA registration completeness.

Author’s Address

Shintaro Yamakawa (editor)
AILEX LLC / VeritasChain Standards Organization
1-10-8 Dogenzaka, Shibuya-ku, Tokyo
150-0043
Japan
Email: info@ailex.co.jp
URI: https://ailex.co.jp