Nick Clark · Published April 25, 2026 · PDF

How to Attribute AI-Generated Content to Its Source for Licensing

If you run a generative system and you want to pay the creators whose work influenced an output, you need attribution that is computable rather than guessed. This guide describes an architecture for deriving attribution from logged consultation events and structural proximity, so a generated artifact yields auditable attribution records from which compensation obligations can be computed. It describes an architecture disclosed in PCT International Application No. PCT/US26/28630, not a shipping library, and it builds on the Content Anchoring inventive step.

What You Are Building

You are building an attribution and compensation layer for a generative system: a way to answer, for any given output, "which registered source works contributed to this, and by how much," and to turn that answer into a payment obligation you can defend in an audit.

This is the problem facing anyone operating a generative model over licensed or licensable material: a retrieval-augmented image or text generator, a music model consulting a reference corpus, a platform that ingests creator work into a training set

and wants to compensate those creators. Regulators, rightsholders, and your own accounting all want the same thing: a number per creator that traces to something real, not a hand-wave about "training data influence."

The architecture described here computes that number from two ingredients: a deterministic log of the reference artifacts a generation event actually consulted, and a structural proximity score between each consulted artifact and the output. No central rights registry sits in the path, and you do not need to open up the model to inspect its weights.

Why the Obvious Approaches Fall Short

The common approaches each break at a structural point, and it is worth being precise about where.

Model attribution methods try to estimate how much a given training example influenced an output by analyzing gradients, activations, or influence functions. These require white-box or gray-box access to the model, are expensive to compute, and produce estimates rather than records. A rightsholder auditing a model they did not train cannot run them at all.

Watermarking and metadata tagging embed an identity signal in the content or a sidecar file. As the filed disclosure notes, watermarks are removable through transcoding, cropping, or generative reconstruction, and metadata records are decoupled from content structure and require persistent external storage. An attribution scheme that depends on a signal an adversary can strip is not a licensing foundation.

Central rights registries put a single authoritative database in the resolution path. This works until the registry is unreachable, disputed, or simply not trusted by every party who needs to rely on the attribution. It also binds identity to registration events

rather than to the content itself.

Location or hash-based identifiers (URLs, file hashes, paths) are invalidated by mutation, format conversion, resolution change, or compression, so they fragment identity across the very derivatives you need to attribute. A cryptographic hash changes completely when one pixel changes; it cannot tell you that two artifacts are near-relatives.

The structural gap common to all of these: attribution is treated as something you reconstruct after the fact, from either the model internals or a fragile external label. The approach below instead makes each consultation a first-class, logged event at generation time, and grounds identity in the content's own structure.

The Architecture

The disclosed approach has four load-bearing pieces. Each traces to the filed specification.

1. Structural identity for every artifact. Each content artifact is assigned a unique identifier (UID) derived deterministically from its internal variance and structural features rather than from storage location, filename, or transmission metadata. The disclosure extracts a multi-axis variance vector (nine dimensions organized into X, Y, and Z axes capturing energy distribution across scale, frequency compaction, and gradient-orientation structure) from a normalized scalar field. Because two UIDs encode positions in a continuous variance space, cosine similarity between them is directly computable without decoding a binary digest. The same pipeline is disclosed for images, audio (via mel-spectrogram), text (via a token-frequency scalar field), video, and binary objects, so a single similarity operator spans modalities.

This matters for attribution because it gives you a proximity measurement between any output and any registered source that survives format conversion and rescaling within defined thresholds, while diverging predictably as content-altering mutations occur.

2. A consultation event logger. This is the heart of the licensing story. The disclosure specifies a consultation event logger that records, for each generation event that consults a reference artifact through retrieval-augmented generation or structured neighborhood resolution, a deterministic consultation record. Each record comprises the variance-derived UID of the consulted artifact, the governing policy object, the variance proximity score between the consulted artifact and the generated output, and a timestamp.

The disclosure states plainly what this buys you: these consultation events become "computable attribution events from which compensation obligations may be derived under policy-declared schedules," rendering "creator payment computable from consultation event logs rather than from approximations of training data influence." Attribution attaches to a governed consultation event, not to a reverse-engineering of model weights.

3. An alias record that carries payment routing. A creator registers a human-readable alias against the UID of their work. Per the disclosure, that alias record carries a compensation routing field encoding a payment address, a compensation schedule reference, and a jurisdictional scope. The payment address is deliberately payment-infrastructure-agnostic: it may designate a bank routing number, a digital payment service identifier, a blockchain wallet address, or an internal ledger account. The routing field says where; the schedule says how much.

4. Attribution and compensation computation. The disclosed attribution computation module aggregates consultation events per consulted artifact and computes an attribution weight as the product of consultation frequency and mean variance proximity score. The compensation computation module multiplies that

attribution weight by the schedule rate drawn from the compensation schedule reference and produces a payment obligation record. A payment routing module credits the obligation to the payment address in the alias record under the declared jurisdictional scope, and a compensation audit log appends each obligation record.

The compensation schedule reference is itself a versioned, signed, machine-evaluable document. The disclosure lists per-consultation flat rates, proximity-weighted rates (higher cosine similarity yields proportionally higher payment), category-specific rates, and jurisdiction-specific rates. Schedule versioning is what makes a computation reproducible: you can replay the obligation from the schedule version that was in effect at the consultation timestamp.

Two properties fall out of this design. First, the compensation audit log is verifiable against the consultation event log: any party can confirm that payment obligations are consistent with the corresponding consultation events and the schedules in effect, without access to model weights, training logs, or proprietary platform data. Second, a related mechanism in the disclosure, the model output provenance fingerprint, lets a party who holds only an output and access to the training corpus anchor index measure structural proximity to specific training artifacts without model introspection, membership inference, or the model operator's cooperation. That is the audit path for outputs whose consultation was not logged at generation time.

How to Approach the Build

You are implementing this yourself. Here is a sensible order.

Step 1, build the variance extraction and UID derivation pipeline. Implement the multi-axis variance vector extraction over a normalized scalar field for whatever modality you handle first (images are the worked example in the disclosure). This is the

foundation; every later step consumes UIDs and proximity scores from it. Confirm your UIDs are stable across format conversion and rescaling within your chosen thresholds before moving on.

Step 2, register sources and their aliases. For each source artifact you want to be able to compensate, compute its UID and create an alias record. An illustrative record shape, faithful to the disclosed fields:

```
// Illustrative only, not a shipping schema
aliasRecord = {
  alias:        "studio/frost-series-03",
  uid:          <320-bit variance-derived UID>,
  compensation: {
    paymentAddress:  <bank | wallet | ledger | service id>,
    scheduleRef:     <versioned signed schedule id>,
    jurisdiction:    <scope>
  }
}
```

The three pre-conditions the disclosure names must all hold for compensation to be computable: the source has a registered UID, the creator has registered an alias with a compensation routing field, and generation happens in an environment that logs consultation events.

Step 3, instrument generation to emit consultation records. Wherever your generation pipeline retrieves or resolves reference artifacts (your RAG retrieval step, or a structured nearest-neighbor lookup), emit a consultation record at that point. Compute the variance proximity score between the consulted artifact's variance vector and the generated output's variance vector using the cosine similarity operator. An illustrative record:

```
// Illustrative only
consultationRecord = {
  consultedUid:  <uid>,
  policyVersion: <governing policy object id>,
  proximity:     cosineSimilarity(consultedVec, outputVec),
  timestamp:     <t>
}
```

This is the step most likely to be wrong in practice: if your generator consults sources through a path you do not instrument, those consultations never enter the log and never earn attribution. Treat "every consultation is logged" as an invariant you test for, not a hope.

Step 4, aggregate and compute. Periodically (or per settlement window), aggregate consultation records per consulted UID. Compute attribution weight as consultation frequency times mean proximity score, exactly as disclosed. Resolve each UID's alias record, pull the referenced schedule version, multiply attribution weight by the schedule rate, and write a payment obligation record.

Step 5, route payment and append to the audit log. Resolve the payment address format and execute the credit through the appropriate interface, then append the obligation record (consulted UID, attribution weight, schedule version, amount, address credited, jurisdiction, timestamp) to an append-only compensation audit log. Keep the log independently verifiable against the consultation log; that verifiability is the point.

Step 6, add the after-the-fact audit path. For outputs you did not generate, or consultations you did not log, implement the model output provenance fingerprint query: extract the output's variance vector, query the training corpus anchor index for artifacts within a proximity radius, and produce a signed provenance record of matched

UIDs and cosine scores. This gives rightsholders and regulators an attribution path that does not depend on your cooperation, which is often exactly what makes the scheme credible.

What This Does Not Give You

Be clear-eyed about the boundaries.

This is an architecture, not a drop-in library. There is no package to install and nothing that "just works." You implement every component above yourself, and the illustrative snippets are sketches of disclosed data shapes, not runnable code. The approach is disclosed in a patent filing; it is not a benchmarked or productized system, and this guide reports no performance numbers because the filing establishes an architecture, not measured results.

The attribution weights and proximity scores are structural signals, not legal determinations. The disclosure is explicit that lineage edge weights and memorization proximity scores do not constitute legal determinations of authorship, ownership, or infringement; they are inputs that may inform licensing, attribution display, and compensation. The legal meaning you assign to them is your policy choice, encoded in your signed policy and schedule objects.

Compensation is only computable when the three pre-conditions hold. If a source has no registered UID, or a creator never registered an alias with a routing field, or generation happens outside an environment that logs consultations, there is no consultation-derived obligation to compute. The provenance-fingerprint audit path can still measure proximity after the fact, but it measures structural proximity, not a logged consultation event.

Finally, the quality of your attribution is bounded by the quality of your instrumentation and your proximity thresholds. Un-instrumented retrieval paths produce silent gaps; poorly calibrated similarity thresholds either miss real derivatives or over-attribute. These are engineering responsibilities the architecture places on you, not guarantees it hands you.

Disclosure Scope

The approach described in this guide is disclosed in PCT International Application No. PCT/US26/28630. This guide is educational: it explains the architecture and how a developer might approach building it, drawing solely on that filing. It is not a warranty, a specification of a released product, or an offer of software, and nothing here should be read as a legal determination of authorship, ownership, infringement, or the compensation owed in any particular case. Implementers are responsible for their own build, their own policy and schedule objects, and their own legal and regulatory compliance.

Content Anchoring (/content-anchoring)

All 40 steps → (/inventive-steps)

Computable identity for media. Provenance from structural variance.

PCT/US26/28630 (/patents/pct-us26-28630)

PRIMARY TECHNICAL DISCLOSURE

- Content Anchoring: Computable Identity for Media That Changes (/articles/content-anchoring-computable-identity-for-media-that-changes)

SECONDARY TECHNICAL

- Multi-Axis Variance Vector Extraction: Nine Dimensions of Structural Content Identity (/articles/content-anchoring/variance-vector)

- Quadrant Decomposition: Spatial Sub-Region Fingerprinting for Partial Similarity Detection (/articles/content-anchoring/quadrant-decomposition)
- 320-Bit UID Construction: Multi-Segment Hashing for Negligible Collision Probability (/articles/content-anchoring/uid-construction)
- Structure Signature: Background-Invariant Matching Through Gradient-Only Descriptors (/articles/content-anchoring/structure-signature)
- Constellation Signature: Geometry-Invariant Matching Across Crop, Scale, and Occlusion (/articles/content-anchoring/constellation-signature)
- Five-Band Variance Classification: Content Routing by Structural Complexity (/articles/content-anchoring/variance-classification)
- Variance Saturation-Governed Cache Eviction: UID Density Replacing Static TTL (/articles/content-anchoring/cache-eviction)
- Multi-Root Composite Lineage Graphs: Provenance Through Variance Vector Similarity (/articles/content-anchoring/composite-lineage)
- Multi-Modal Content Identity: Unified Pipeline Across Image, Audio, Text, and Video (/articles/content-anchoring/multi-modal-identity)
- Rights-Grade Pre-Release Admissibility: Policy Evaluation Before Content Commitment (/articles/content-anchoring/pre-release-admissibility)
- Training Corpus Governance: Verifiable Lineage From Training Data to Model (/articles/content-anchoring/training-corpus-governance)
- Consultation Event Logging: Deterministic Records of Every Generation Reference (/articles/content-anchoring/consultation-logging)
- Model Output Provenance Fingerprint: Structural Proximity Without Model Access (/articles/content-anchoring/output-provenance)
- Creator Attribution and Compensation Routing: Payment From Consultation Lineage (/articles/content-anchoring/creator-attribution)
- Adversarial Robustness and Deepfake Detection: Content Identity as Detection Substrate (/articles/content-anchoring/adversarial-robustness)
- Client-Side Execution Architecture: Privacy-Preserving Variance Computation on Device (/articles/content-anchoring/client-side-execution)
- UID Resolution Query Protocol: Distributed Lookup Across Anchor Node Networks (/articles/content-anchoring/uid-resolution)
- Orientation Canonicalization: Rotation-Invariant Processing Through Gradient Normalization (/articles/content-anchoring/orientation-canonicalization)
- Cross-Band Resolution Pathfinding: Traversal Between Variance Bands Under Mutation (/articles/content-anchoring/cross-band-resolution)

- Identity by Position: Media as a Third Navigable Space (/articles/content-anchoring/identity-by-position)

APPLICATIONS · GENERAL

- Forbidden-Content Blocking at Upload and Generation Time: Pre-Release Exclusion Against Signed Policy (/articles/content-anchoring/forbidden-content-blocking)
- Structural Provenance for Software Supply Chains: Binary and Firmware Identity Independent of SBOM Metadata (/articles/content-anchoring/software-supply-chain-provenance)
- Rights-Grade Generative AI: How to Pay Creators, Exclude Forbidden Content, and Prevent Infringement Before Release (/articles/content-anchoring/rights-grade-generative-ai)
- Deepfake Detection by Structural Provenance: Verifying Synthetic Media Without Watermarks (/articles/content-anchoring/deepfake-provenance)
- Creator Economy Attribution Without Platform Intermediaries (/articles/content-anchoring/creator-attribution-economy)
- Verifying Source Photos and Video in the Newsroom: Content Anchoring for Journalism (/articles/content-anchoring/journalism-verification)
- Detecting Image Manipulation and Proving Figure Provenance in Research Publications (/articles/content-anchoring/academic-research-integrity)
- Content Anchoring for Legal Evidence Chains (/articles/content-anchoring/legal-evidence-chain)
- Content Anchoring for Insurance Claims Evidence (/articles/content-anchoring/insurance-claims-evidence)
- Content Anchoring for Real Estate Documentation (/articles/content-anchoring/real-estate-documentation)
- Art Authentication and Provenance Verification with Content Anchoring (/articles/content-anchoring/art-authentication)
- Detecting Screenshot and Recapture Fraud in Identity-Document KYC With Structural Content Identity (/articles/content-anchoring/identity-document-kyc-recapture)

APPLICATIONS · SPECIFIC

- C2PA vs Content Anchoring: Attached Provenance or Content-Intrinsic Identity? (/articles/content-anchoring/c2pa)
- Google SynthID Alternative: Content-Intrinsic Identity Beyond Watermarking (/articles/content-anchoring/google-synthid)
- Beyond Shutterstock: Content-Intrinsic Identity That Survives Re-Encoding and Cropping (/articles/content-anchoring/shutterstock)
- Spotify Alternative for Music Provenance: Structural Content Identity Beyond the ISRC Database (/articles/content-anchoring/spotify)

- Getty Images Alternative for Provenance: Structural Content Identity Beyond Metadata (/articles/content-anchoring/getty-images)
- Adobe Stock vs Structural Content Identity: Licensing Records Are Not Content Identity (/articles/content-anchoring/adobe-stock)
- YouTube Content ID vs Content Anchoring: Matching Against a Database, or Identity in the Content Itself (/articles/content-anchoring/youtube-content-id)
- Audible Magic Alternative: Structural Content Identity Beyond Database-Matched Fingerprinting (/articles/content-anchoring/audible-magic)
- Digimarc vs Structural Content Identity: Watermarks Are Added, Not Intrinsic (/articles/content-anchoring/digimarc)
- Irdeto vs Structural Content Identity: DRM Protects the Channel, Not the Payload (/articles/content-anchoring/irdeto)
- Truepic alternative: capture-time provenance versus structural identity derived from the artifact itself (/articles/content-anchoring/truepic)
- Microsoft PhotoDNA vs structural content identity: hash-matching known images versus screening artifacts before release (/articles/content-anchoring/microsoft-photodna)
- Pex alternative: structural content identity vs enrolled fingerprint matching (/articles/content-anchoring/pex)

Content Anchoring overview → (/content-anchoring)