Nick Clark · Published July 2, 2026 · PDF

How to Verify Content Authenticity Without a Central Registry

If you need to tell whether a piece of content is authentic, derived, recaptured, or synthetic, but you cannot depend on a central registry, an embedded watermark, or an enrollment step, this guide describes an architecture for doing it structurally. The approach is disclosed in PCT International Application No. PCT/US26/28630 and is presented here as an architecture you build yourself, not as a shipping library. Its home inventive step is the Content Anchoring inventive step.

What You Are Building

You are building a way to answer a hard question about a digital artifact: is this content what it claims to be? Concretely, given an image, audio clip, document, video, or binary blob, you want to determine whether it is a known registered artifact, a derivative of one, an orphan with no traceable lineage, or a probable screen recapture or synthetic generation. You want to do this without standing up a central registry that every participant must trust and query, without embedding a watermark that transcoding can strip, and without an enrollment step that requires the original creator to have registered the artifact through your system before it existed.

The people who have this problem are content platforms verifying uploads, model operators checking outputs before release, rightsholders auditing whether generated content is structurally proximate to their work, and anyone building in a federated or intermittently connected environment where a single authoritative registry is either a liability or simply unreachable. This guide walks through the architecture disclosed in PCT International Application No. PCT/US26/28630 so a skilled developer can understand how to approach the build. It is an architecture, not a package you install.

Why the Obvious Approaches Fall Short

The conventional tools each carry a structural limitation, and it is worth stating them accurately rather than knocking down straw men.

Cryptographic hashes such as SHA-256 or MD5 are excellent at answering "are these two byte streams identical?" but that is a different question from "is this the same content?" Any re-encoding, resolution change, format conversion, or lossy compression changes the bytes and therefore the hash, so a hash cannot recognize a resized or transcoded version of the same artifact, and it tells you nothing about similarity or derivation.

Watermarking and metadata tagging embed an identity signal in the content stream or a sidecar record. These work when the signal survives, but watermarks are removable through transcoding, cropping, or generative reconstruction, and metadata records are decoupled from the content structure and require persistent external storage to stay attached.

Blockchain-based registration anchors ownership records to a distributed ledger through hash-based proofs. This gives you tamper-evident records, but it binds identity to key-pair ownership rather than to content structure, requires global consensus and transaction cost, and cannot detect semantic similarity between related or derivative objects.

Perceptual hashing (difference hash, average hash, phash) gets closer by producing a signature from a downsampled image, but the filed disclosure notes that these fixed-width binary outputs lack multi-scale structural analysis and a continuously scaled similarity score, so they are limited for lineage tracing and derivative attribution.

The common thread: each approach identifies content by something attached to it (a byte layout, an embedded mark, a ledger entry, a coarse binary digest) rather than deriving identity from the internal structure of the artifact itself. That is the gap the disclosed architecture targets.

The Architecture

The core idea in the filing is a content identity computed post-hoc from the artifact itself. Nothing is embedded, no enrollment is required, and no central registry is needed. Everything below traces to PCT/US26/28630.

Normalize to a scalar field. Whatever the modality, you first reduce the artifact to a bounded two-dimensional scalar field of normalized values. For raster images this is a grayscale floating-point representation using perceptual luminance weighting (about 0.299 red, 0.587 green, 0.114 blue) normalized to [0, 1]. The disclosure describes modality-specific paths that all converge on the same downstream pipeline: audio becomes a normalized mel-spectrogram, text becomes a token-frequency grid (TF-IDF weighted) optionally blended with byte-level variance, video is handled at the frame level plus a clip-level temporal delta vector, and binary objects are reshaped into a near-square matrix of per-window byte statistics.

Extract a multi-axis variance vector. The pipeline subdivides the scalar field into three nested grids (8x8 coarse, 16x16 medium, 32x32 fine) and computes per-cell variance, aggregated into mean and standard-deviation-of-variance per scale. From these it builds a nine-dimensional vector across three axes: an X axis encoding cross-scale energy distribution (slope, curvature, asymptotic fine-scale energy), a Y axis

encoding frequency compaction behavior (rate of change of variance spread, spread factor, variance-floor convergence), and a Z axis encoding structural phase persistence from a gradient-orientation histogram over eight angular bins spanning zero to pi, canonicalized so the dominant bin sits at index zero. The Z axis yields a horizontal-vertical orientation bias, a diagonal-axial bias, and a stability coefficient.

Build the identifier. The nine-dimensional vector is combined with per-quadrant vectors. The artifact is normalized to a 256x256 canonical canvas, optionally rotated to a canonical orientation, split into four non-overlapping quadrants, and each quadrant is independently run through the same extraction and hashed (X and Y quantized at 1/32, Z at 1/8, using overlapping FNV-variant hashes). The four quadrant hashes are sorted lexicographically for rotation invariance, then combined with the global hash by a multi-segment FNV-64 combiner into a 320-bit UID. The key property the filing emphasizes: this UID encodes a position in a continuous variance space, so cosine similarity between any two UIDs is directly computable without decoding a fixed binary digest. That is what makes similarity, derivation, and lineage first-class operations rather than an afterthought.

Route by variance band, not by address. The global variance value places each UID in one of five variance bands (the disclosure gives thresholds: below 0.02, up to 0.06, 0.12, 0.22, and at or above 0.22). Anchor nodes declare governance over one or more bands. Because any node that knows an artifact's variance value can compute which band governs it, a resolution query can be routed to the right anchor cluster without consulting a central directory. Anchors within a band coordinate through a quorum-based consensus described in the filing (trust-weighted, asynchronous, lineage-preserving) rather than through a single authority.

Answer authenticity questions structurally. On top of this, the filing describes the verification signals you actually want:

- *Lineage and orphan detection*. Query the anchor network for registered parent UIDs within a configured slope-continuity radius. If none fall within the radius, the artifact is classed as structurally unanchored, an orphan with no provable lineage. The filing is careful that orphan status is not proof of fraud, but such an artifact cannot satisfy a policy that requires verifiable provenance.
- *Screenshot recapture detection*. When a display renders an image and a camera or capture device re-captures it, sub-pixel geometry and optics introduce a characteristic signature: elevated energy in horizontal and vertical orientation bins relative to diagonal bins. This shows up as a systematically elevated Z-axis horizontal-vertical bias. A recapture classifier compares that bias to a policy-calibrated threshold. Notably, this requires no reference to the original and no corpus lookup; it reads only the candidate's own structure.
- *Synthetic content detection*. Compare the candidate's variance vector against a slope-band-indexed statistical model of known synthetic-content variance profiles. Falling inside the synthetic distribution and outside the authentic one for the category yields an elevated synthesis probability score.
- *Commitment-boundary admissibility*. Rather than moderating after release, the architecture interposes a pre-release check at the "commitment boundary" (any irreversible or externally visible effect: publish, deliver, return from an API, admit to a training corpus). A candidate is checked against a governed exclusion corpus by cosine similarity; if it exceeds a policy-declared threshold it is rejected, regenerated, or escalated. Because this runs over variance-derived UIDs rather than GPU embedding inference, the filing states it can run client-side at generation time.
- *Consultation-event attribution*. When a generative system consults a reference artifact (through retrieval or structured neighborhood resolution), a deterministic consultation record is logged: the consulted UID, the governing policy version, the variance-proximity score, and a timestamp. This makes attribution computable from logged events rather than from reverse-engineering model weights.

How to Approach the Build

A realistic order of implementation, given that you are building this yourself:

1. **Implement the extractor first, and pin your canonical parameters.** The 256x256 canvas, the three grid scales, the eight-bin gradient histogram, the luminance weights, and the quantization steps are the contract. Two nodes must produce the same UID for the same artifact, so treat these as a versioned specification. Start with raster images because the other modalities reduce to the same scalar-field pipeline.
2. **Verify stability before you verify identity.** The claimed value is that the UID is stable under format conversion, resolution rescaling, and moderate lossy compression while diverging under semantic edits. Build a test harness that takes source artifacts, applies transcodes and re-encodes, and confirms cosine similarity stays high, then applies object insertion, cropping, and style transfer and confirms it drops. If your quantization is too fine, compression noise will fragment identity; if too coarse, distinct content will collide. This tuning is the real work.
3. **Make cosine similarity the primitive.** An illustrative interface sketch, faithful to the disclosure and clearly not a shipped library:

```
# illustrative only
uid       = encode(normalize(artifact))     # 320-bit, carries a 9-dim va
band      = variance_band(uid.global_variance)
score     = cosine(uid.variance_vector, other.variance_vector)  # in [-1,
```

   Every higher-level decision (identity, derivative, orphan, exclusion) is a threshold over this score, so get it right and reuse it everywhere.

4. **Define resolution modes as thresholds.** The filing names four: identity resolution (exact or near-exact match), derivative resolution (similarity between the continuity threshold and the identity threshold), orphan resolution (empty match set), and conflict resolution (overlapping lineage claims routed to adjudication). Encode these thresholds in your policy objects, not in code, so they are auditable and versioned.
5. **Build the anchor layer as band-scoped governance, not servers.** Route a query by computing the band from the candidate's variance, then dispatch to the anchors governing that band, with adjacency-ordered referral to neighboring bands when the primary band misses. Keep queries stateless: each carries the candidate UID, a policy-scope identifier, and a timestamp. The disclosure notes this rides over HTTP, WebSockets, WebRTC, or delay-tolerant mesh, so do not couple it to one transport.
6. **Transmit UIDs, not artifacts.** A significant property is that the raw artifact never has to leave the client. Compute the UID locally, evaluate against a locally cached, signed exclusion-corpus fragment and a signed policy object, and send only the UID and the decision upstream. This is what enables client-side and disconnected evaluation.
7. **Layer the verification signals on last.** Orphan detection and recapture detection are cheap wins because recapture needs only the candidate's own Z-axis bias. Synthetic-content detection needs you to maintain a reference distribution of synthetic variance profiles, which you update as new generators appear. Commitment-boundary admissibility needs a governed exclusion corpus and signed policy objects in place first.

What This Does Not Give You

Be honest with yourself about the boundaries.

This is an architecture disclosed in a patent filing, not a benchmarked, productized, or drop-in library. There is no SDK to install and no promise that it "just works." You implement the extractor, the anchor layer, the policy format, and the thresholds yourself, and the quality of your result depends heavily on the parameter tuning described above.

The filing does not state accuracy numbers, false-positive rates, or performance benchmarks, and neither should you. The recapture, synthetic, and memorization signals are described as probability scores and structural proximity signals, and the disclosure is explicit that they are signals informing further review, not legal determinations of authorship, infringement, or ownership. A high memorization-proximity or lineage-attribution score is a structural fact, not a verdict.

Orphan status means "no registered lineage in your governed corpus," which is only as meaningful as the corpus is complete; a legitimately new original will also read as an orphan. Stability holds within defined thresholds, so a heavy enough transformation is, by design, a different artifact. And the exclusion, similarity, and memorization checks are all relative to a governed corpus and signed policy objects you must build and maintain; the architecture supplies the mechanism, not the corpus or the policy content.

Disclosure Scope

The structural content identity approach described in this guide is disclosed in PCT International Application No. PCT/US26/28630. This guide is educational: it explains an architecture and how a developer might approach building it, and it is not a warranty, a benchmark, a shipping product, or an offer of software. Every mechanism described here traces to that filing; where the filing is silent, this guide makes no claim. Third-party technologies referenced for context (cryptographic hashing, watermarking, blockchain registration, perceptual hashing, DNS, and PKI) are described neutrally and are the property of their respective owners.

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)