IPFN

International Pinned File Number



Technical White Paper

A web-native identifier for creative work.

Controlled registration. Publicly verifiable evidence.

Version 2.0 | 1 November 2026

A Light Across the Canyon | Kilian Liptrot

LENS-yc.3pfn.xyz | ipfn.net/experience

IPFN Technical White Paper

A controlled registration system with publicly verifiable evidence

Version 2.0 | 1 November 2026. Implementation examined at repository commit `d4319713ffb00b0b9fe7cfc3025fe675a362d07c`. Observations are dated separately from implementation descriptions. This paper documents the system and explains how to examine its evidence; it does not grant authority to issue IPFNs or a license to the underlying software.

Abstract

An image can travel farther than the name of the person who made it. IPFN addresses that practical identification problem by giving registered creative work an identifier that works like a web address. A reader can use that address to find the recorded claimant and inspect the evidence associated with the registration. The system connects controlled identifier issuance, an exact-file SHA-256 fingerprint where available, an IPFS certificate, and a daily ledger whose fingerprint is committed to Base mainnet. Many records share one blockchain transaction.

The distinction between issuance and verification is central. Authorized registration authorities create records within the IPFN system. Anyone with the published artifacts can independently compare their bytes and blockchain commitment. Public verification does not make issuance permissionless, and a valid cryptographic comparison does not decide authorship or copyright ownership.

This paper defines the actors, objects, identifier forms, registration states, ledger serialization, anchor checks, and verification outcomes in the current implementation. It also identifies the trust that remains with the claimant, registration authority, namespace operator, storage providers, and chain-access provider. A production example demonstrates a 113-entry ledger and an independently rehashed original photograph. The evidence supports exact-file correspondence and record integrity; it does not establish when the photograph was taken, who created it, or that the service will remain available indefinitely.

1. The problem and the contribution

Artists cannot reliably attach their names to work web-wide.

Ever seen a photograph in the wild and wondered who took that image? A caption can disappear when the image is copied. Embedded metadata can disappear during an export. A gallery visitor may see a print without knowing where to find the photographer. Copyright can exist while the practical route back to the creator is missing.

IPFN works like an ISBN for images: an assigned identifier connects the work to a registration that another person can open. The analogy concerns identification and registration authority. Neither an ISBN nor an IPFN grants copyright. For IPFN, the reader can go further and compare the registered file and ledger evidence. [S1]

Blinding Visuals Press encountered this problem while preparing visual works for publication and commercial use. The resulting file chain coordinates existing technologies around one registered claim. DNS makes the address reachable; SHA-256 fingerprints exact bytes; IPFS gives the certificate a content address; the daily ledger groups registrations; and Base records the ledger commitment outside the operator's editable database.

The contribution is this controlled registration and verification arrangement, not a new hash algorithm, storage network, blockchain, or consensus mechanism. The paper makes no claim that the combination is

the first possible system of its kind. Its technical value is assessable through the objects it binds, the checks it exposes, and the failures it distinguishes.

2. Position among existing systems

System	Principal function	Relationship to IPFN
ISBN	Identifies a publication's edition and format for the book trade.	The useful institutional analogy: assigned identifiers and registration agencies. IPFN is separate and claims no affiliation with the International ISBN Agency. [S1]
DOI	Persistent object identification, metadata and resolution, administered through registration agencies.	A strong example of managed identifier stewardship. IPFN's present verification path additionally specifies its file fingerprint, certificate and daily blockchain commitment; it is not a DOI implementation. [S2]
ISCC	Content-derived codes for metadata, similarity and exact-instance identification.	IPFN's serial is assigned, not computed from the content. Similarity codes may assist discovery; they do not replace the registered claim or issue its identifier. No ISCC integration is claimed here. [S3]
C2PA	Signed provenance assertions bound to media, with defined validation and trust mechanisms.	A different evidence layer. Preserving an original can preserve embedded credentials, but the current IPFN service does not thereby validate or issue C2PA credentials. [S4, C12]
IPFS	Organizes and retrieves content-addressed data.	Used for the certificate. A CID identifies content structure; it does not identify the author, assign an IPFN, or fund preservation. [S5]
Base	An Ethereum layer 2 network with publicly inspectable transactions.	Holds the daily commitment on chain ID 8453. Each work does not receive its own Ethereum mainnet transaction. [S6]
W3C DID	Identifiers and method-specific operations for subjects, documents and verification methods.	IPFN uses its controlled DNS namespace rather than DID syntax or a DID method. Public checking does not require permissionless identifier creation. [S7]

These comparisons identify different jobs, not a contest in which one identifier must replace every other standard. A book can have an ISBN and an IPFN. A photograph can have embedded provenance credentials and a separately registered IPFN. Actual integrations must be demonstrated individually; a comparison is not a conformance certificate.

3. Actors and objects

The **claimant** is the person or organization whose claim is recorded. The **submitter** operates the account and supplies the file and metadata; that person may act for someone else. Account access establishes a relationship to the service, not an independent finding of legal identity or authorship.

The **registration authority** accepts the registration under IPFN's rules. The implementation maps image types `LENS` and `ART` to ARTSSN, and publication types `BKS` and `MAG` to Bookend Press. The **registry operator** maintains the assigned identifiers and their records. The current public operating identity is Blinding Visuals Press with the International IPFN Agency. The **anchor operator** controls the configured transaction-signing key. Storage and DNS operators keep the referenced material reachable. The **verifier** reads evidence and performs comparisons; it needs no power to register a work or send a transaction. [C1, C2, C8]

Object	Precise meaning
Creative work	The photograph, artwork or publication about which a claim is made. It is not identical to one file encoding.
Work file, `W`	A particular sequence of bytes submitted or supplied for checking. A JPEG export and its edited version are different byte sequences.
Registration, `R`	The system's assigned IPFN, object reference and recorded claimant information, with links to available evidence.
Work digest, `hW`	`SHA256(W)`, recorded for an image. It identifies exact bytes, not an artistic idea.
Certificate, `C`	The published certificate bytes containing the registration assertions. Historical receipts can include a text heading before their JSON.
Certificate CID, `cC`	The IPFS address for the certificate's encoded content graph, not the photograph's raw SHA-256.
Daily ledger, `L`	The exact canonical JSON byte sequence containing an ordered batch of registration entries.
Ledger digest, `hL`	`SHA256(L)`, carried in the Base transaction's data.
Anchor, `T`	The specific transaction committing the ledger digest, together with its receipt and block evidence.
Catalog presentation	Current title, claimant display, caption and preview shown on the website. It can change independently of already committed bytes.

The IPFN identifies a registration concerning a work; it is not the work digest itself. Current image records support a `content_digest` scope. Current book and magazine records can be `catalog_assertion` records with no digest of the complete publication files. A schema for future publication packages exists, but a provisional schema is not proof that a signed, complete-file publication pipeline is operating. [C2, C3]

4. Identifier assignment and resolution

New ordinary identifiers use `TYPE-SERIAL.3pfn.xyz`. Types are `LENS`, `ART`, `BKS` and `MAG`. The documented generator draws serial characters from `abcdefghjkmnpqrstuvwxyz23456789`, starts at two characters, increases length after each 1,000 unsuccessful attempts, checks existing identifiers and banned patterns, and stops after 10,000 attempts. The registry's case-insensitive unique index protects the identifier key; the serial is not a secret or an authentication token. This is an implementation account of issuance, not an invitation to mint names outside the registry. [C1]

Parsing is deliberately broader than generation. It accepts case variations, legacy `TYPE.SERIAL.3PFN.XYZ` forms, historical `1pfn.xyz` workspaces, and the legacy `BOOK` type in dot form. A designated exception accepts the inaugural `LENS-1.3pfn.xyz`; ordinary one-character serials are not thereby authorized. `normalizeIpfnQuery` also extracts an identifier from supported URL paths or hostnames. A string passing the parser is only well formed: the registry must still contain the registration. [C1]

Legacy acceptance preserves references already printed or shared. It does not mean every historical workspace is currently served by every DNS deployment; syntax acceptance and live reachability require separate checks. The current `3pfn.xyz` web resolver uses ordinary DNS and HTTPS. ENS is not required by the examined resolution path. A wildcard response supplies a page and Open Graph fields for compatible previews. Platforms choose whether and when to render those previews. A preview finds the claim; it does not independently hash the file. [C4, S8]

The design treats issued identifiers as continuing references. The registry key's uniqueness is demonstrated in source, but this paper does not claim a fully published lifetime non-reassignment and succession policy. That administrative commitment must be maintained alongside the namespace. Repeating a digest under another record would not, by itself, resolve a competing authorship claim.

5. Registration: from a submitted image to evidence

The submitter enters an authorized registration workflow. For an image, the original file is stored and its digest is derived from the stored bytes on the server. The immediate finalize path calculates `serverSha256` and rejects a differing browser-supplied value with `digest_mismatch`, HTTP 422. The archival path streams and hashes the original before completing processing and rejects `archive_digest_mismatch`. The browser value is a cross-check, not the authoritative source. These are source-observed safeguards; this paper did not replay a live upload to prove deployment of every branch. [C5]

Publishing assigns or recovers the registration and connects the image fingerprint to it. Certification then produces a receipt and pins its exact bytes through IPFS. The examined certificate path carries fields including `IPFN`, `AUTHOR` or `AUTHORS`, `TYPE`, `TITLE`, `PUBLISH_DATE`, `PUBLISHER`, `REGISTRATION_AUTHORITY`, `EDITION` and `AUTHENTICITY_CODES`. For the image specimen, `AUTHENTICITY_CODES.SHA_256` records the work digest. These names express certificate assertions; a field called `AUTHOR` is not a court's finding. [C6]

“Certificate” here means the registration artifact. It must not be confused with an X.509 identity certificate, a qualified electronic timestamp, or a claimant's digital signature. The historical receipt examined below is bound through its CID and ledger inclusion; no separate claimant-signature validation is demonstrated.

Pinning is an ongoing storage operation. IPFS content addressing lets another party check retrieved bytes, while maintained pins and reachable services make retrieval possible. The certificate can remain identifiable even when a gateway cannot serve it. That situation is unavailable evidence, not a cryptographic mismatch. [S5]

State or event	What it establishes
Registered	A registry entry exists; pinning and anchoring may remain incomplete.
Pinned	The operator recorded a certificate CID. Retrieval and CID validation are separate verifier checks.
Included in a volume	A volume is linked to the record; independent inclusion requires finding the entry in the committed ledger.
`pending`	The volume awaits its anchor operation.
`sending`	A transaction identity has been prepared and retained; this does not establish a successful receipt.
`anchored`	The worker recorded validated inclusion. Finality is a further observation.
`failed`	The anchor operation failed or was refused; a registration link does not override that result.

The public lookup vocabulary derives `anchored`, `ledger_pending`, `link_failed` and `unledgered` separately. Test evidence remains marked as test, even when the underlying transaction is on Base mainnet. There is no single “verified” state that proves every relationship in the file chain. [C7]

6. FileChain: deterministic batches and public commitment

The daily builder selects eligible unledgered, pinned records before its cutoff. This permits catch-up inclusion; a volume's date must not be read as every work's creation date. Production eligibility uses the recorded registration-time boundary, so repinning an old test record does not automatically turn it into production evidence. [C8]

The current ledger schema is `1.3.0`. Its top-level fields, in order, are:

`schema_version`, `volume_label`, `ledger_date`, `window_start`, `window_end`, `previous_base_tx_hash`, `previous_ledger_sha256`, `base_chain_id`, `ipfn_count`, `standards`, `publisher`, `ipfns`.

The `standards` object orders `timezone`, `locale`, `protocol_language`. Each entry orders `ipfn`, `ipfn_type`, `claimant`, `pinned_cid`, `content_sha256`, `last_pinned_at`, `target_url`. Entries are sorted by the IPFN string using the source's lexicographic comparison. `ipfn_count` equals the array length. `content_sha256` can be null for a publication catalog assertion. Titles are not present in this ledger schema. [C9]

The builder requires explicitly zoned timestamps and normalizes `window_start`, `window_end` and entry pin times to `YYYY-MM-DDTHH:mm:ss.sssZ`. `ledger_date` remains a date. It constructs the fixed-order object, applies ECMAScript `JSON.stringify(ledger, null, 2)`, appends one line-feed character, and encodes UTF-8. It does not apply an additional Unicode normalization pass or use an interchangeable “any canonical JSON” formatter. ECMAScript string escaping remains relevant to reproduction. [C9, S9]

For verification, preserve the downloaded bytes and hash them directly. Parsing and reformatting first can change the digest. Rebuilding the bytes in another language requires the same field order, sorting, timestamp normalization, JSON escaping, whitespace and final newline. This white paper documents that implementation profile; it does not claim a separately certified cross-language conformance suite.

Schema `1.3.0` links to the preceding successfully **built** ledger using `previous_ledger_sha256`, whether that earlier volume is pending, sending or anchored. Selection excludes failed or archived volumes, then orders

by ledger date, sequence and creation time. A predecessor lookup error refuses the build rather than silently manufacturing a genesis. The older `previous_base_tx_hash` remains a compatibility reference to an earlier anchored transaction; it is not the authoritative linear ledger link. [C9]

The canonical storage key is create-only: identical existing bytes permit a no-op, while different bytes under the same key cause refusal. Historical bytes are not rewritten to the newest format. Version `1.0.0` had the initial entry fields; `1.1.0` added descriptive and digest fields; `1.2.0` removed title; `1.3.0` added the previous-ledger digest and strict timestamp normalization. [C9]

The transaction payload is UTF-8 text:

```
FILECHAIN V2 VOL=<volume> SHA256=<hL> PREV=<previous hL or GENESIS>
```

Historical missing-version and `1.0.0`-`1.2.0` ledgers use `FILECHAIN VOL=<volume> SHA256=<hL>`. The payload helper chooses V2 for other declared versions. A verifier still needs to recognize the ledger schema before interpreting its fields; selecting a payload format is not permission to assume unknown schemas have current semantics. [C10]

The worker checks chain ID 8453 and the served ledger digest. It prepares a signed zero-value self-transaction, retains the exact signed bytes and transaction hash before broadcast, and uses a conditional database transition to claim the volume. Recovery can reconcile or rebroadcast that same transaction identity rather than silently sign a replacement. Receipt validation checks the chain, requested hash, configured sender and destination, value, payload and canonical block. Finality is promoted separately after fresh receipt checks and a finalized-head observation. These are implemented controls, not a claim of live fault injection or an independent Base-node audit. [C11]

7. Independent verification

A public verifier needs the registration identifier `I`, exact ledger bytes `L`, transaction identity `T`, certificate CID and certificate bytes or blocks, and access to Base transaction, receipt and block data. The work bytes `W` are optional but necessary for an exact-file result. An auditor also needs a separately established authorization context: which sender was authorized to anchor for the operator at the relevant time.

The procedure below defines an evidence report for this paper. It is not a claim that the public lookup API already returns this complete report or that a public dispute workflow has been commissioned.

1. **Locate and classify.** Normalize `I` using the accepted syntax and obtain its registry reference. Retain source URLs, retrieval times, environment and declared schema. A missing record is `not_found`; an unreachable service is `unavailable`. Do not convert either into a finding about copyright.

2. **Hash before parsing.** Calculate `hL = SHA256(L)` over the exact download. Parse the ledger under its declared supported schema. Check field types, count, identifier uniqueness and the selected entry. Report absent or duplicate inclusion explicitly; a hash match cannot repair a malformed or ambiguous record.

3. **Read the transaction.** Confirm the RPC chain is 8453, retrieve the requested transaction and receipt, and compare transaction identities. A missing receipt is pending or unavailable until resolved; a reverted receipt fails the successful-anchor check.

4. **Compare the commitment.** Decode the transaction data as UTF-8 and construct the exact expected V1 or V2 payload from the ledger's own schema, volume label, calculated digest and previous digest. Compare the complete strings, not a substring. Confirm the containing block agrees with the receipt and the canonical block at that height. Report whether the observed finalized head has passed it. Record provider and observation time. [C10, C11, S6]

5. **Check anchor authority separately.** Compare the transaction's sender, self-destination and zero value with the approved operator context for that period. A matching payload from an unknown address can establish a public commitment, but not an authorized IPFN anchor. If the historical authorization context is unavailable, report `authority_unestablished`; do not invent one from the address appearing in the transaction.

6. **Check the certificate.** Validate its CID using the indicated codec, multihash and complete encoded content graph, or a known importer profile that reproduces it. A raw SHA-256 of a gateway file is not generally its CID. Preserve any receipt heading when checking bytes. For the demonstrated small CIDv0 certificate, the verifier reconstructs a single DAG-PB UnixFS File without links or optional metadata. Unknown profiles are `unsupported`, not a successful check. [S5, S10]

7. **Bind the objects.** After certificate byte validation, parse its supported receipt format and compare its IPFN and available work digest with the selected ledger entry. Compare claimed roles as recorded assertions. Check the previous-ledger hash recursively when making a chain-history claim; one volume's success does not prove completeness of the operator's entire history.

8. **Compare the work.** If `W` and a valid recorded work digest exist, calculate `SHA256(W)` and compare them. If either is absent, return `not_checked` or `not_applicable` for a catalog assertion. Never substitute the website's compressed preview for the claimed original.

Return a report with separate results for `record`, `schema`, `ledger_bytes`, `inclusion`, `transaction`, `authority`, `finality`, `certificate`, `record_binding`, `work_file` and `history`. Each check records `match`, `mismatch`, `pending`, `unavailable`, `unsupported`, `not_checked` or `not_applicable` as appropriate, with the observed values. A published dispute notice, if supplied by a responsible authority, is recorded separately with its source. Absence of such a notice is not proof that the claim is undisputed.

The three principal relationships are distinct: `SHA256(W) = hW`; certificate graph validation yields `cC`; and `SHA256(L) = hL` in the examined transaction. A certificate match does not establish a work-file match. A ledger match does not establish authorized issuance. None of these operations determines legal ownership.

8. Trust boundaries and adversaries

The operator controls intake, issuance and presentation. Published commitments reduce the ability to change committed evidence without detection; they do not remove operator responsibility. A reader must know which authority's claim is being inspected and which signing identity the authority recognizes.

The worker enforces its configured notary address. In the revised demonstration, independent transaction reconstruction and signature recovery identify a sender matching the operator-declared anchor address. That is a concrete authorization reference, distinct from a complete historical key policy. The examined evidence does not establish a public, time-versioned key-authorization and rotation history. This paper supplies no invented revocation periods, continuity guarantees or retrospective key approvals. A signing-key compromise can produce technically valid transactions containing false new assertions. An independently maintained authorization history and incident procedure are institutional requirements, not properties supplied by SHA-256. [C11, C13, E2]

Threat	Check or boundary
Another person registers a stolen photograph	Preserve the identity assurance actually obtained and permit evidence review. An earlier timestamp cannot make a false authorship claim true.
Certificate, ledger or source bytes are substituted	Recompute the relevant digest or CID and compare the complete binding; return mismatch.

Threat	Check or boundary
An attacker posts the right payload from another address	Separate payload agreement from anchor authorization.
A preview or edited export is presented as the original	Hash the independently supplied file; identify derivatives separately.
A gateway, DNS name or registry becomes unavailable	Return unavailable for affected checks; a chain commitment cannot reconstruct missing files.
A registration is omitted from a ledger	Require its entry for inclusion. One valid batch does not prove that all eligible registrations were included.
A provider returns inconsistent chain data	Cross-check transaction, receipt and canonical block; additional providers or an independently operated node reduce dependence on one RPC account.
Current metadata differs from the old assertion	Preserve committed bytes and label the current catalog separately. Corrections need an explicit history; changing a webpage does not rewrite an old certificate.

Public claimant names and certificate content can carry personal data. Retention, corrections, lawful publication and international processing require an operating policy outside the cryptographic comparisons. An immutable commitment is not a waiver of data protection obligations. The companion governance paper addresses that institutional work.

9. Measured evidence and limits

The 31 October 2026 production volume contains 113 registrations in 43,758 canonical JSON bytes. Its SHA-256 is `94c4ba61af14f74dd15bdc3cd10d210b055671a3172ae9ca58153d49a9f9f9ba`. One Base transaction, `0x25b1532e64f3af4d363cf21df71ee56b189a5644876f9139ec5c310ba57f8251`, committed that digest and the preceding ledger digest. The retained check observed a successful receipt in block 53,351,386, timestamped `2026-11-01T00:15:19Z`, below the finalized head returned by the public RPC. [E1]

The revised specimen is *A Light Across the Canyon*, `LENS-yc.3pfn.xyz`, recording Kilian Liptrot as claimant. A separate read-only download of its 1,591,834-byte original JPEG produced `938e065ecc9b7ff0b40ccd604d1c9ddcb4c0f70e03b35e5d89efdd749c1279f2`, matching the ledger's work digest. The certificate CID recorded for this entry is `QmQEWUWBh6afkwW8ByiBUyCpWrmG9RLxo8QqUHf9R5rYsE`. Independent reconstruction of the signed transaction reproduced its hash; signature recovery yielded `0x20ce5b8aaF38c647A42CfC6dC038a12A12648F6e`, agreeing with the RPC sender and the operator-declared anchor address returned by the authenticated FileChain state endpoint. This establishes file correspondence and agreement with that declared signing identity. It does not reconstruct the historical upload, establish who took the photograph, or independently certify legal key ownership and every historical authorization change. The companion demonstration retains the artifact and transaction checks. [C13, E2]

The revised reproduction also reconstructed the certificate CID, checked unique ledger inclusion and agreement between the displayed claim, certificate and ledger, and compared the exact transaction input, successful receipt, block references and reported finality. In-memory changes to the original JPEG, certificate and ledger failed their comparisons. No public record was altered. Source availability is a separate outcome from a byte mismatch. [E2]

These observations demonstrate batching and reproducible comparisons. They do not measure registration throughput, median or tail anchor latency, long-term availability, recovery success rate or total cost per

retained work. A fixed-size digest commits to a growing batch, but storage, hashing, indexing, retrieval and support still grow. No unlimited-capacity, subsecond verification, or permanent-free-storage claim follows.

Further acceptance tests should change the original, substitute a certificate, alter a ledger, supply an unauthorized sender, present pending and reverted transactions, withhold a gateway, and challenge the claimant. Expected results are the specific file mismatch, certificate mismatch, ledger mismatch, authority failure, pending/failure, unavailable or review state. These are defined test cases until executed, not a retrospective claim that every case passed.

10. Operation and development

Blinding Visuals Press and the International IPFN Agency can maintain a controlled registration system while allowing public examination of evidence. Institutional archives, publishers and developers can consume references, retain evidence and perform read-only checks without becoming authorized issuers. Appointment of additional registration authorities remains a separate administrative and contractual decision. This paper neither announces such appointments nor changes software rights.

The repository includes perceptual-hash and OpenCLIP paths that may help identify related images. Exact hashes and similarity models answer different questions. Code presence is not a benchmarked web-wide copying service. A plagiarism-detection beta needs a defined corpus, transformation tests, error rates, review thresholds and an appeal process before its accuracy is advertised. [C12]

The next institutional step is a bounded pilot using real works, retained originals and independently performed checks. It should also test corrections, disputed claims, service failure and evidence export. Continued usefulness depends on namespace stewardship, pinning, accessible ledger copies, signing-key accountability and funding. The DOI system provides a relevant lesson: persistence requires an organization and its continuing commitments as well as a technical identifier. [S2]

IPFN's claim is concrete: a controlled registration can connect a creative work to a named claimant and published evidence that another person can inspect. The system becomes more credible when its checks return the precise result supported by those bytes, including when the answer is pending, unavailable or unproved.

Figure A. The system, following one photograph

How IPFN records a claim

Follow one photograph from the claimant to the evidence anyone can inspect.



A Light Across the Canyon

Kilian Liptrot

Recorded claimant

One transaction can cover many registered works.

Storage and ongoing access still require an operator.

01

Fingerprint the file

SHA-256 produces a code from the exact file bytes.

02

Assign the IPFN

LENS-yc.3pfn.xyz opens the work's registration.

03

Save the certificate

IPFS holds the identifier, claim and file fingerprint at a content-based address.

04

Anchor the ledger

Many certificates enter one ledger. Its hash is recorded on Base, an Ethereum Layer 2.

What does opening the IPFN show?

The work, its registered claimant, and links to the fingerprint, certificate and ledger evidence.

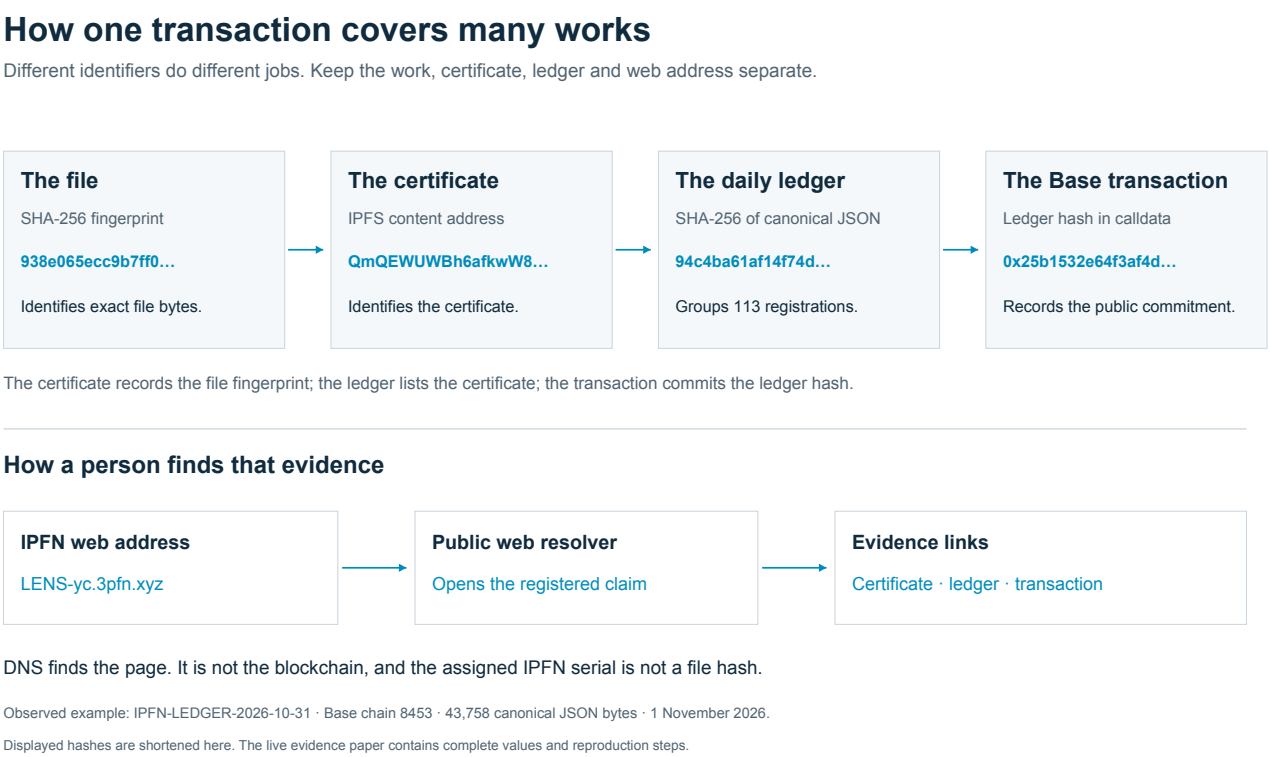
The web address finds the registration. The fingerprints let someone check particular bytes.

The evidence supports a recorded claim; registration does not decide copyright ownership.

Diagram uses the real A Light Across the Canyon original JPEG. Technical values and proof are in the accompanying paper.

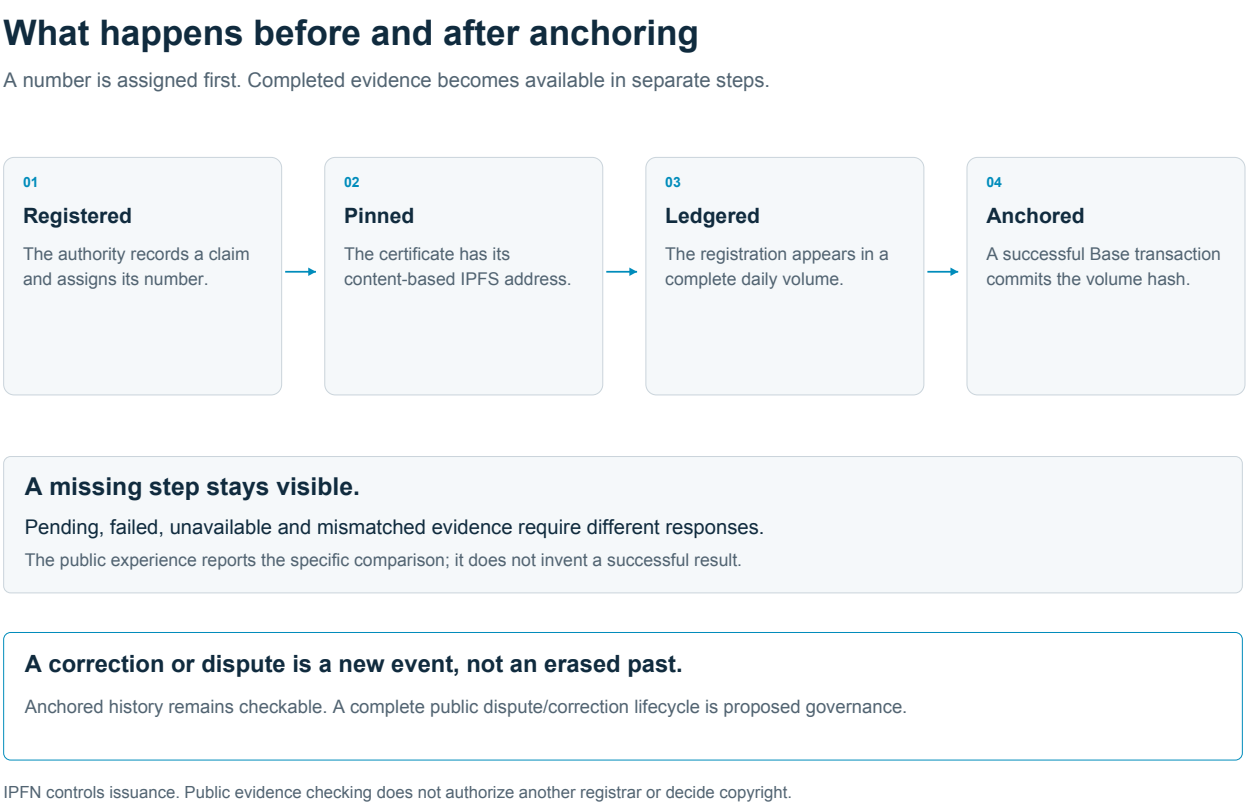
The registered number opens the web record directly. The certificate, ledger and Base commitment supply different evidence that a reader can inspect. The blockchain does not assign the number or perform DNS resolution. ARTSSN is the registration authority for this selected photograph.

Figure B. The three byte comparisons



The original file fingerprint, certificate content address and ledger fingerprint identify different byte sequences. The transaction commits the complete ledger hash. Following the web address finds the record; comparing the downloaded bytes tests the stated relationships.

Figure C. Registration and evidence states



Registration, pinning, ledger inclusion and anchoring occur separately. A public reader reports the completed comparison rather than combining them into an unsupported ownership guarantee. A complete correction/dispute process requires the governance described in the companion proposal.

References and implementation evidence

Primary specifications were checked on 1 November 2026. Comparisons do not imply affiliation, integration or certification.

S1. International ISBN Agency, *What is an ISBN?* Edition/format identification, trade use and distinction from copyright protection. https://www.isbn-international.org/index.php/content/what-isbn/10

S2. DOI Foundation, *What is a DOI?* and *What Are Registration Agencies?* Persistent resolution, metadata, controlled agency participation and continuity responsibilities. https://www.doi.org/the-identifier/what-is-a-doi/ ; https://www.doi.org/the-community/what-are-registration-agencies/ . Handbook landing and version of record: https://www.doi.org/the-identifier/resources/handbook

S3. ISCC, *Specification*, describing ISO 24138:2024 and the distinction between the published core and later extensions. https://iscc.codes/specification/

S4. C2PA, *Technical Specification 2.4*, sections 9-15, content binding, claims, signatures, trust and validation. https://spec.c2pa.org/specifications/specifications/2.4/specs/C2PA_Specification.html

S5. IPFS, *Content Identifiers* and *Pin files*. https://docs.ipfs.tech/concepts/content-addressing/ ; https://docs.ipfs.tech/how-to/pin-files/

S6. Base, *eth_chainId*, *eth_getTransactionReceipt*, and derivation specification. https://docs.base.org/base-chain/api-reference/ethereum-json-rpc-api/eth_chainId ; https://docs.base.org/base-chain/api-reference/ethereum-json-rpc-api/eth_getTransactionReceipt ; https://docs.base.org/base-chain/specs/protocol/consensus/derivation

S7. W3C, *Decentralized Identifiers*, identifier, controller, method and resolution model. https://www.w3.org/TR/did-core/

S8. *The Open Graph protocol*, page metadata used by compatible consumers. https://ogp.me/

S9. ECMAScript, `JSON.stringify`. https://tc39.es/ecma262/multipage/structured-data.html#sec-json.stringify . SHA-256 definition: NIST, FIPS 180-4. https://csrc.nist.gov/pubs/fips/180-4/upd1/final

S10. IPFS, *UnixFS*, DAG-PB file encoding and content graph structure. https://specs.ipfs.tech/unixfs/

All C references below are pinned to repository commit `d4319713ffb00b0b9fe7cfc3025fe675a362d07c`; line numbers refer to that exact version. These are implementation evidence, not claims that every path was exercised on production during this review.

C1. `src/lib/ipfn/format.ts`, lines 13-121; `migrations/MIGRATION_IPFN_DASH_SEPARATOR.sql`, lines 56-94; `migrations/20260603_ipfn_registry_citext_unique.sql`, lines 252-258. Parsing, issuance algorithm and unique-key definition. The historical format document's old ledger-schema description is superseded by current code.

C2. `src/lib/cert/registrationAuthority.ts`, lines 21-70; `src/lib/filechain/verifyResolver.ts`, lines 90-129. Authority mapping, evidence scope, and current catalog projection.

C3. `src/lib/filechain/publicationManifest.ts`, lines 3-24 and 86-104. Provisional publication-package model and explicit unimplemented artifact/signing boundary.

C4. `cloudflare/proof-router-worker/worker.js`, `fetchProofMeta`, `renderProofHtml` and wildcard route; `src/middleware.ts`, host-based resolution. Live reachability and platform rendering are separate observations.

C5. `src/app/api/uploads/finalize/route.ts`, lines 137-211; `src/lib/intake/workerEntry.ts`, lines 255-280; `src/lib/intake/archiveFile.ts`, lines 10-49. Server-derived original digest, browser cross-check and archival custody.

C6. `documentation/hetzner/embed_api.py`, lines 462-517; `src/lib/cert/runIpfsPreview.ts`, lines 58-175. Pinned receipt construction and separately rendered certificate fields; not evidence of a detached claimant signature.

C7. `src/lib/ipfn/anchorStatus.ts`, lines 19-46; `src/lib/filechain/verifyResolver.ts`, lines 31-74. Display vocabulary and persisted anchor states.

C8. `src/app/api/cron/filechain-build-ledger/route.ts`, lines 440-445 and 516-559; `src/lib/filechain/productionEligibility.ts`, lines 1-22. Candidate cutoff, predecessor lookup and production boundary.

C9. `src/lib/filechain/buildLedger.ts`, lines 59-135, 140-237, 256-296 and 303-349. Schema history, fixed-order construction, timestamp rules, previous-volume selection and byte serialization.

C10. `src/lib/filechain/anchorPayload.ts`, lines 3-56. Historical V1 and current V2 payload selection. The worker's matching implementation is `cloudflare/anchor-worker/src/pure.ts`.

C11. `cloudflare/anchor-worker/src/index.ts`, lines 489-582, 653-825, 830-946 and 1279-1408; `cloudflare/anchor-worker/src/pure.ts`, `receiptValidationDecision`. Same-transaction recovery, receipt identity checks, configured signer and staged finality. No complete public historical key-rotation policy is established by these paths.

C12. `src/lib/fingerprint/dispatch.ts`, lines 137-143; `src/lib/fingerprint/driftCompare.ts`; `src/lib/publishPipeline.ts`, `createImageFingerprint`; `src/app/(ipfn)/ipfn/learn/page.tsx`, Originals, exports and C2PA section. Exact original digests, derivative fingerprints and the boundary of C2PA behavior.

C13. `src/app/api/bvpress/filechain/state/route.ts`, line 86, `notary_address`. The authenticated state response is the operator's declared address reference; it does not publish a complete historical key-rotation policy.

E1. Retained first-edition read-only verifier and provider responses: `docs/evidence/p531/technical/verify_live_evidence.py`, lines 56-109, and `evidence/verification-summary.json`, observation `2026-11-01T16:53:16.836790+00:00`. Public canonical ledger: https://images.blindingvisuals.com/filechain-ledgers/production/IPFN-LEDGER-2026-10-31.json . Transaction: https://basescan.org/tx/0x25b1532e64f3af4d363cf21df71ee56b189a5644876f9139ec5c310ba57f8251

E2. Revised demonstration, *A Light Across the Canyon*, `LENS-yc.3pfn.xyz`: https://ipfn.net/verify/LENS-yc.3pfn.xyz . Original-byte observation and the revised certificate/transaction checks are retained with the accompanying revision-2 evidence demonstration. The photograph is a real registered work; it is not a generated illustration.