ODNCA-STD-010 · ORD-META-1: On-Chain Content Discovery

The discovery layer for on-chain content: OP_RETURN metadata records, trust levels, and name-bound canonicalization

Status: v1.0 — Adopted 14 September 2026 (public review 30 August – 13 September 2026, GOV-001 §5)
Companion documents: ODNCA-STD-003 (SNS-NAME-1), ODNCA-STD-004 §6 (state commitments), ODNCA-STD-005 / STD-011 (the ord:// scheme), ODNCA-STD-002 (living-registry pattern)

Chain scope: BSV only.

The key words MUST, MUST NOT, SHOULD, MAY are to be interpreted as in RFC 2119.

Naming note: the `-1` suffix is the standard's major version, per ODNCA house convention (SNS-NAME-1, WEB3-ACTIONS-1), and equals the payload field `"v": 1`. Compact profiles for byte-constrained chains, if ever wanted, would be separate standards; they are out of scope here by decision.

§1 · Problem & design premise

Inscription payloads may be compressed, encrypted, minified or binary; a search spider cannot be required to fetch and parse every payload. ORD-META therefore defines a dedicated **discovery layer**: a small, always-readable metadata record in an OP_RETURN output that describes a content inscription, points to it, and states how its payload is stored. Spiders scan only this layer; fetching content is optional (§5).

Premises inherited from the founding discussion (7 May 2026) and ratified here: 1. OP_RETURN is the discovery layer; payloads are opaque to spiders. 2. Metadata MUST be readable: plaintext JSON, or gzip; **never encrypted**. 3. One TX can carry multiple inscriptions, so a bare txid is not a valid pointer; and when metadata travels in the same TX as its single content inscription, the pointer MUST be omitted (atomic binding, §4). 4. BSV-only: no byte budget, no binary packing. BSV permits multiple OP_RETURN outputs per TX, which resolves the batch question by construction: a batch TX carries one ORD-META output **per page**.

§2 · Envelope (normative)

An ORD-META record is a single OP_RETURN output:

```
OP_FALSE OP_RETURN <"ordmeta"> <"1"> <payload>
```

- Push 1: the ASCII protocol tag `ordmeta`.
- Push 2: the ASCII major version `1`.
- Push 3: the payload — UTF-8 JSON, or RFC 1952 gzip of that JSON (detected by magic bytes `1f 8b`). Encrypted or otherwise unreadable metadata payloads are nonconforming and MUST be ignored.

The payload's `p` and `v` MUST equal the envelope's tag and version; a mismatch makes the record nonconforming. (The envelope duplicates them so scanners can filter without JSON parsing; the payload carries them so a stored record remains self-describing.)

A TX MAY contain any number of ORD-META outputs. Each content record (`op` = `"meta"`, the default when `op` is absent) describes exactly one content inscription; a reset record (§6a) describes none.

§3 · Payload schema (normative)

```
{
  "p": "ordmeta",
  "v": 1,
  "op": "meta",
  "type": "text/html",
  "enc": { "compression": "gzip", "encryption": null },
  "site": "name.web3",
  "path": "/about",
  "title": "About us",
  "desc": "Author-written description for result snippets.",
  "excerpt": "Verbatim plaintext fragment taken from the content body.",
  "lang": "en",
  "keywords": ["ordinals", "naming"],
  "category": "technology",
  "author": "name.web3",
  "ts": 1755850000,
  "hash": "<sha256 hex of the target inscription's stored payload bytes>",
```

```
  "license": "CC-BY-4.0",
  "parent": "<txid>_<vout>",
  "ref": "<txid>_<vout>"
}
```

Required **for `op = "meta"`**: `p`, `v`, `type`, `title`, `ts`, `hash` — and `enc` whenever the target payload is not stored as plain bytes of `type`. All other fields optional.

Required **for `op = "reset"`**: exactly `p`, `v`, `op`, `site`, `ts`; all other fields MUST be absent (§6a). A reset record carrying any other field **defined by this standard** is nonconforming; fields unknown to this standard are ignored on reset records as everywhere else.

Unknown fields MUST be ignored by indexers (forward compatibility). Field rules:

- **op** — `"meta"` (default when absent) or `"reset"` (§6a).
- **type** — MIME type of the target content as intended for consumption (what the payload *is* once retrieved and decoded).
- **enc** — how the payload is stored: `compression` (`"gzip"`, `"br"`, or null) and `encryption` (scheme identifier such as `"aes-256-gcm"`, or null). Absent `enc` means plain. This is the retrieval instruction: a client MUST be able to decide from the record alone how to open the target (or that it cannot, when encrypted).
- **hash** — sha256 over the target inscription's payload bytes **exactly as stored on-chain** (before any decompression or decryption). This makes L1 verification reproducible for every record, including encrypted content.
- **site** + **path** — binds the content to an SNS name and route; together they form the canonical address `ord://<site><path>`. `site` MUST satisfy SNS-NAME-1; `path` MUST start with `/`. **Scope note:** although the `ord://` scheme also loads dotless OpNS names, `site` in ORD-META v1 is deliberately SNS-only — the L2 holder proof (§5) verifies against the committed name state, and OpNS state is not yet committed. A future revision may extend `site` to OpNS names once it is. When `site` is set and `path` is absent, `path` defaults to `/` (the site root) — for the canonical address and for §6 canonicalization alike.
- **desc** — author-written description, ≤ 160 Unicode codepoints.
- **excerpt** — plaintext fragment from the content body, ≤ 300 codepoints. It SHOULD be a verbatim substring of the (decoded) content; when the content is readable, indexers MAY verify this and SHOULD demote records whose excerpt is fabricated. `desc` and `excerpt` are distinct signals (author claim vs body evidence) and MUST NOT be merged by producers.
- **title** ≤ 120 codepoints, **keywords** ≤ 10 items ≤ 32 codepoints, **lang** = BCP-47, **category** = free text v1 (a curated registry MAY follow as living registry, per the STD-002 pattern).
- All length limits are in Unicode codepoints. Indexers MUST truncate over-limit values rather than reject the record, and truncation MUST NOT split a codepoint.
- **author** — an SNS name (verifiable, see §5) or free text (display-only).
- **ts** — author-claimed unix time; block time is authoritative for ordering.
- **parent** — the ORD-META record this one supersedes (versioning chain). Here `<txid>_<vout>` identifies the **OP_RETURN output** carrying the prior record — not an inscription — so §4's resolution rules do not apply to `parent`, and an unresolvable `parent` does not make a record nonconforming (it is informative history, §6).
- **license** — SPDX identifier where possible.

§4 · Pointer rules (normative — applies to `op = "meta"` records only)

Reset records carry no target; this section does not apply to them (§6a).

This standard assumes the 1Sat ordinals model: **one inscription per output**, so `<txid>_<vout>` identifies an inscription. A future profile MAY add a third index component for models with multiple inscriptions per output; conforming ORD-META-1 records never need it.

- **Same-TX binding (preferred):** if the ORD-META output is in the same TX as its target and the TX carries exactly one content inscription, `ref` MUST be omitted; the target is that inscription. If the TX carries more than one content inscription, every ORD-META output in it MUST carry an explicit `ref` — `"_<vout>"` (txid omitted) for a target in this TX, or `"<txid>_<vout>"` for a target elsewhere. No positional or output-order inference is permitted.
- **Cross-TX pointer:** `ref` = `"<txid>_<vout>"` of the target inscription output.
- A record whose target cannot be resolved is nonconforming and MUST be ignored.

§5 · Trust levels & anti-spam (normative for indexers)

The levels below classify **content claims** (`op` = `"meta"`). Records with `op` = `"reset"` are not content claims: §4 and levels L0/L1 do not apply to them; they are **name-bound** when the holder proof below succeeds via its input branch (a reset has no target, so the sat branch cannot apply), evaluated immediately before the reset's transaction in chain order (§6). Only name-bound reset records have any effect (§6a).

Indexers classify every conforming content record:

- **L0 — scanned:** envelope valid, payload parses, target resolves; the content itself has **not** been fetched and `hash` is unverified. This is the cheap scan mode promised by §1: an indexer MAY operate entirely at L0 and verify lazily, on demand, or by sampling.

- **L1 — content-verified:** the indexer fetched the target's stored bytes and `hash` matches. A **verified mismatch** makes the record nonconforming (it describes nothing) and it MUST be removed from the index; mismatch is not a mere demotion.
- **L2 — name-bound:** the record is L1, `site` is set, and the **holder proof** succeeds: the ORD-META TX spends at least one input from the holder script of `site`, **or** the target inscription resides on the sat that carries the name. Holder state is evaluated against the SNS state **immediately before the record's transaction**, intra-block position included (chain order, §6). A record placed by a holder who transfers the name later — even later in the same block — remains name-bound ("valid when placed"); a record placed after the transfer, even in the same block, is not.

Verification note: the ODNCA-STD-004 §6 state commitments anchor state at commit heights. TX-precise holder state between commitments is obtained by replaying chain data forward from the nearest committed snapshot under the published indexer rules — a deterministic replay, not a direct lookup. Implementers of the holder proof must plan for this.

Two consequences indexers must understand before implementing: an L0-only indexer can list and search records but can never serve canonical `(site, path)` answers — canonicalization (§6) operates on name-bound records, and reaching L2 for content claims requires the L1 fetch. And on encrypted content, L1 attests **integrity only** (the stored bytes are what the author committed to), not descriptive accuracy: `type`, `title`, `desc`, `excerpt` and `lang` remain unverifiable claims there, and indexers SHOULD treat them accordingly in ranking. `enc` itself is a producer claim tested by **no** trust level — a record may omit or misstate it and still reach L1, since `hash` covers the stored bytes. Clients MUST treat decode failures as content errors and fail rendering; they MUST NOT trust the record's description over the bytes.

Only L2 records may claim a `(site, path)`. This is the standard's central economic property: claiming a name's search presence requires the name — scarce, on-chain, committed — not merely bytes. L0/L1 records without `site` are indexed on their own merits and MUST NOT be attributed to any name.

§6 · Canonicalization & versioning (normative)

Chain order means: block height, then TX order within the block, then output order within the TX. It is the single ordering used throughout this standard — for canonicalization, for the holder proof (§5), and for reset supersession (§6a).

Per `(site, path)`: among L2 records, the latest in chain order wins. `parent` chains are informative history; indexers SHOULD expose them but MUST NOT let a parent claim override chain order. Records without `site` are canonical only for their own target inscription.

Multiple records MAY describe the same target inscription — including within one TX (per §4, in a single-inscription TX every ORD-META output omits `ref` and binds to it). The latest in chain order is the target's canonical record. Producers wanting parallel variants (e.g. languages) SHOULD use distinct `(site, path)` addresses rather than competing records for one target.

§6a · Name transfers & reset records

A transfer of `site` does **not** retroactively invalidate L2 records placed by the previous holder — they were valid when placed, and archival determinism requires evaluating holdership at the record's chain position. To reclaim a transferred name's search presence without republishing every page, the new holder MAY publish a **reset record**:

```
{ "p": "ordmeta", "v": 1, "op": "reset", "site": "name.web3",
  "ts": 1755850000 }
```

A reset record MUST itself be name-bound per the §5 holder proof — of which only the input branch is applicable, since a reset has no target — placed by the current holder; the L0/L1 machinery does not apply to it. Its effect: all ORD-META records for `site` (every path) **strictly before the reset in chain order (§6)** are marked superseded and MUST NOT be served as canonical. That includes records by the previous holder earlier in the reset's own block, while the new holder's fresh records after the reset — even in the same block — survive, so the §7 pattern (reset first, fresh metadata immediately after) is safe within one block. Multiple resets are allowed; the latest in chain order applies.

§7 · Producer guidance (informative)

- Site builders SHOULD emit one ORD-META output per page **in the same TX** as the page inscription (zero extra transactions, atomic binding, no ref).
- Batch deployments on BSV SHOULD ship N pages + N ORD-META outputs in one TX, each with `ref` = `"_<vout>"` per §4.
- Existing content can be annotated retroactively via cross-TX pointers — but reaches L2 only when placed by the name holder.
- After acquiring a name, publish a reset record first, then fresh metadata for the pages you keep.

§8 · Conformance vectors (to be published with v1.0)

Nine vectors, generated by two independent implementations (the pattern of the published SNS test-vector sets): (1) minimal valid record, (2) full-field record incl. `enc`, (3) gzip metadata payload, (4) same-TX binding without ref, (5) one batch TX with three pages + three records using `"_<vout>"`, (6) an L2 name-bound record incl. holder-proof data, (7) a hash-mismatch record that MUST be rejected at L1, (8) a name-bound reset record superseding earlier paths (holder proof, no L0/L1), (9) two records on one target in one TX with the deterministic canonical selection. Each vector: raw TX hex + expected parse + expected classification. The two implementations MUST be developed independently; vectors 6 and 9 are the designated divergence probes — vector 6 forces the replay logic of §5's verification note, vector 9 the output-order tiebreak within one TX, the two places where independent implementations are most likely to differ. When those two match, the rest tends to follow.

Honesty note on independence: two implementations commissioned by the same steward — even in separate sessions or by separate authors — catch divergence and ambiguity, but do **not** constitute independent third-party verification in the sense the ODNCA trust framework requires. That claim is reserved until an implementer outside the steward's control reproduces these vectors, and ODNCA will label the vectors' provenance accordingly.

§9 · Relationship to other standards

SNS-NAME-1 / ODNCA-STD-003 (name validity), ODNCA-STD-004 §6 (holder verification for L2), the `ord://` URI scheme (ODNCA-STD-005, superseded by STD-011 upon the latter's adoption), STD-002 living registry (future category registry). ORD-META records are not name registrations and not ODNCA attestations; they are third-party content claims that indexers evaluate under this standard.