Independent Submission
Internet-Draft
Intended status: Informational
Expires: 24 March 2026

S. Tomlinson
Lockb0x LLC

20 September 2025

The Lockb0x Protocol: Codex Entries for Verifiable Data Sovereignty
draft-tomlinson-lockb0x-00

Abstract

This document specifies the Lockb0x Protocol, a standards-based framework for creating Codex Entries (machine-readable Controllable Electronic Records) that bind together storage proofs, blockchain anchors, encryption metadata, signatures, and provenance. The protocol enables interoperability across decentralized and cloud storage systems while providing auditability and compliance with legal frameworks such as UCC Article 12.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 24 March 2026.

Copyright Notice

Copyright (c) 2025 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Table of Contents

1. Introduction

The Lockb0x Protocol defines a mechanism for verifiable data provenance and sovereignty across heterogeneous storage backends and blockchains. This Internet-Draft provides a reference specification.

2. Scope

The Lockb0x Protocol defines a verification and provenance layer for digital data. It is *not a storage protocol*; rather, it operates with existing storage and blockchain systems to provide integrity, custodianship, and auditability.

Objectives of the protocol include:

* *Data Sovereignty*: demonstrating control over how and where data is stored.

* *Compliance*: enabling a verifiable chain of custody applicable to frameworks such as UCC Article 12 in the United States and GDPR in the European Union.

* *Cross-Organization Trust*: allowing independent parties to verify shared data without relying on a central authority.

The protocol introduces a *Codex Entry* abstraction that unifies storage proofs, cryptographic signatures, identity bindings, and blockchain anchors in a consistent, verifiable format.

3. Terminology

The key words MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL in this document are to be interpreted as described in [RFC2119].

* *Codex Entry*: A structured, signed data object that encapsulates integrity proofs, storage references, anchors, identity bindings, and optional encryption metadata.

* *Storage Adapter*: A module that interfaces with an external storage system (e.g., IPFS, S3, GCS, Azure, Solid Pod) and produces verifiable proofs of storage.

* *Anchor*: A blockchain transaction or equivalent immutable record that cryptographically attests to the existence of a Codex Entry at a specific point in time.

* *Verifier*: A tool or process that checks the validity of a Codex Entry by verifying signatures, integrity proofs, storage claims, and anchors.

* *Certificate*: A human- or machine-readable document generated from a Codex Entry that provides evidence of integrity, custodianship, and anchoring. Formats MAY include JSON, W3C Verifiable Credentials, or X.509.

* *Identity Context*: Identifiers bound to a Codex Entry that describe the controlling party (individual, entity, or asset) and optional hierarchy fields (org, process, artifact). An additional subject field MAY be included to represent an individual, entity, or asset referenced by the entry.

* *Encryption Metadata*: Information included in a Codex Entry when assets are encrypted, specifying algorithm, key identifiers, and the last_controlled_by field.

4. Requirements Language

This document uses the key words defined in BCP 14 (RFC 2119 and RFC 8174).

The key words MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL in this document are to be interpreted as described in BCP 14 ([RFC2119], [RFC8174]) when, and only when, they appear in all capitals, as shown here.

5. Data Model

The Lockb0x Protocol defines a structured JSON object called a Codex Entry. This is the fundamental unit of the protocol, capturing proofs of integrity, storage, identity, and blockchain anchoring.

The full normative Codex Entry [JSON-Schema] is provided in Appendix B. This section provides only an overview of the object structure and its role in the protocol.

See the specification for required and optional fields, key ownership semantics, and revision chains. UUIDs MUST conform to [RFC9562].

6. Storage Adapters

Storage Adapters translate provider-specific metadata into canonical proofs (e.g., ni-URIs). Supported adapters include IPFS, S3, GCS, Azure Blob, and Solid Pods. Each adapter defines:

* Protocol identifier (e.g., "ipfs", "s3", "solid")

* Method for computing [RFC6920] integrity proof

* Required metadata fields (e.g., region, jurisdiction, provider)

7. Anchoring

Anchoring links a Codex Entry to an immutable, time-stamped record on a blockchain or equivalent ledger. Anchors MUST be created before certificates are issued. Signatures MAY be applied before or after anchoring, but certificates MUST bind both signatures and the anchor. Required fields: chain ([CAIP-2]), tx_hash, hash_alg. Optional: token_id (NFT anchor).

8. Verification

Verification requires recomputing integrity proofs, validating signatures, and confirming anchors. Verifiers MUST implement [RFC8785] JSON canonicalization. Automated verifiers MAY be smart contracts.

9. Certificates

Certificates provide human/machine-readable evidence derived from Codex Entries. Formats: JSON, W3C VC, X.509. Certificates MUST bind to anchor transaction IDs, integrity proofs, and signatures. They MAY include expiry or revocation metadata.

10. Codex Entry JSON Schema

```
{ ... see appendix-b-schema.md for full JSON Schema ... }
```

Figure 1: Codex Entry Schema Reference

11. Example Flows

End-to-end flows for Codex Entries:

1. File -> Codex Entry -> Stellar Anchor -> Verification
2. IPFS Example
3. S3 Example
4. Google Cloud Storage Example
5. Solid Pod Example
6. Multi-Sig Control Flow
7. Revision Chain Flow

(Full example flows are provided in Appendix A.)

12. Security Threat Models

Detailed scenarios include: key compromise, malicious storage providers, replay attacks, hash collisions, false jurisdiction claims, insider threats, denial-of-service, and post-quantum risks. Mitigations include HSMs, multi-sig, canonicalization, crypto-agility, and redundant anchors.

13. Security Considerations

Security considerations for the Lockb0x Protocol are discussed in detail in the "Security Threat Models" section. Implementers should review the threat scenarios and recommended mitigations, including the use of HSMs, multi-signature controls, canonicalization, and crypto-agility. Additional risks may arise from improper implementation or integration with external systems.

14. IANA Considerations

This document makes no request of IANA.

15. Normative References

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

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

[RFC6920] IETF, "Named Information (ni) URI Scheme", RFC 6920, 2013, <https://www.rfc-editor.org/info/rfc6920>.

[RFC8785] Jones, M., "JSON Canonicalization Scheme (JCS)", RFC 8785, 2020, <https://www.rfc-editor.org/info/rfc8785>.

[RFC9562] Leach, P., "Universally Unique IDentifiers (UUIDs)", RFC 9562, 2023, <https://www.rfc-editor.org/info/rfc9562>.

[JSON-Schema] JSON Schema, "JSON Schema: A Media Type for Describing JSON Documents (Draft 7)", 2018, <https://json-schema.org/specification-links.html#draft-7>.

[CAIP-2] Chain Agnostic Standards Alliance, "CAIP-2: Blockchain ID Specification", 2020, <https://github.com/ChainAgnostic/CAIPs/blob/master/CAIPs/caip-2.md>.

Appendix B. Codex Entry JSON Schema

This appendix contains the full JSON Schema for Codex Entries. This schema is normative and defines required and optional fields, their types, and constraints.

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "title": "Codex Entry",
  "type": "object",
  "properties": {
    "id": { "type": "string", "format": "uuid" },
    "version": { "type": "string" },
    "storage": {
      "type": "object",
      "properties": {
        "protocol": { "type": "string" },
        "location": { "type": "string" },
        "integrity_proof": { "type": "string" },
        "jurisdiction": { "type": "string" }
      },
```

```
      "required": ["protocol", "integrity_proof"]
    },
    "encryption": {
      "type": "object",
      "properties": {
        "alg": { "type": "string" },
        "key_id": { "type": "string" },
        "last_controlled_by": { "type": "string" }
      }
    },
    "identity": {
      "type": "object",
      "properties": {
        "org": { "type": "string" },
        "process": { "type": "string" },
        "artifact": { "type": "string" },
        "subject": { "type": "string" }
      },
      "required": ["org", "process", "artifact"]
    },
    "anchor": {
      "type": "object",
      "properties": {
        "chain": { "type": "string" },
        "tx_hash": { "type": "string" },
        "hash_alg": { "type": "string" }
      },
      "required": ["chain", "tx_hash", "hash_alg"]
    },
    "signatures": {
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "alg": { "type": "string" },
          "kid": { "type": "string" },
          "signature": { "type": "string" }
        },
        "required": ["alg", "signature"]
      }
    },
    "previous_id": { "type": "string", "format": "uuid" }
  },
  "required": [
    "id",
    "version",
    "storage",
    "identity",
```

```
                "anchor",
                "signatures"
              ]
            }
```

Figure 2: Codex Entry Full JSON Schema

Author’s Address

Steven Tomlinson
Lockb0x LLC
Email: contact@steventomlinson.dev