Independent Submission
Internet-Draft
Intended status: Informational
Expires: 18 September 2026

L.J. Reilly Jr.
Independent

18 March 2026

Cognitive Trust Stack (CTS): A Framework for Verifiable AI Behavioral Provenance
draft-reilly-cts-00

Abstract

Every artificial intelligence system deployed today operates under behavioral constraints that are unverifiable by any external party. Alignment claims exist as documentation, policy statements, or terms of service -- all mutable, none cryptographically provable. When an AI system causes harm, there is no mechanism to prove what rules it was following. When an organization claims its AI is safe, there is no standard by which that claim can be independently verified. This is the AI behavioral provenance problem, and no existing framework solves it.

The Cognitive Trust Stack (CTS) establishes that the alignment state of an AI system at any point in time MUST be a verifiable fact, not an assertion requiring trust. CTS defines a complete, implementable framework for declaring, anchoring, enforcing, and verifying AI behavioral constraints through a five-layer architecture combining: (1) a declarative alignment schema formatted to IETF Internet-Draft standards, (2) archival permanence via Digital Object Identifier (DOI) registration, (3) cryptographic temporal anchoring via Bitcoin blockchain timestamping using the Dual-Layer Digital Permanence (DLDP) methodology [REILLY-REM], (4) runtime retrieval injection for active constraint enforcement, and (5) independent third-party verification.

CTS does not replace existing AI alignment techniques. It provides the missing accountability layer that sits above them -- making alignment a cryptographically provable fact rather than an unverifiable claim.

The full CTS specification, reference implementation, schema, and provenance manifest are published at Zenodo DOI 10.5281/zenodo.19097169. The priority of this framework is cryptographically anchored in Bitcoin block 941168 (2026-03-18), with SHA-256 hash: e915b5162422281e1c0185c9e2ee faf74b7f539996b878cb1e69e10533f24ac2

The OpenTimestamps proof file (CTS_Whitepaper_v1.0.docx.ots), included in the Zenodo record, provides independent cryptographic verification of existence prior to block 941168.

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 18 September 2026.

Table of Contents

1. Introduction

1.1. The Broken Promise of AI Alignment

Artificial intelligence systems -- particularly large language models (LLMs) and autonomous AI agents -- are now deployed at global scale across healthcare, finance, legal services, education, infrastructure, and government. Each of these deployments carries an implicit or explicit promise: that the AI system behaves according to a declared set of rules, constraints, and ethical commitments.

This promise is, in every current deployment, unverifiable.

Organizations declare alignment through blog posts, terms of service, model cards, and internal policy documents. Regulators reference compliance frameworks. Researchers

publish alignment techniques. But not one of these mechanisms produces a cryptographically verifiable record that answers the fundamental question:

> "What behavioral rules was this AI system actually operating under at this specific moment in time -- and how can anyone independently prove it?"

No existing standard answers this question. No existing framework makes AI alignment a provable fact rather than an asserted claim. The result is a global AI governance environment built on trust without verification -- the precise condition that cryptographic and distributed systems engineering has spent decades learning to eliminate in other domains.

CTS is the missing verification layer. It establishes that AI behavioral provenance -- the cryptographically verifiable record of what constraints an AI system was operating under at a given time -- is not only necessary but achievable today using existing infrastructure, without requiring changes to underlying AI systems, new hardware, or proprietary tooling.

1.2. The Four Failures of Current AI Governance

The absence of a behavioral provenance standard creates four compounding failures that no existing framework resolves:

Auditability Failure: Organizations cannot produce cryptographic proof of the alignment rules their AI systems enforced at deployment. Internal documentation is mutable. Vendor attestations require trust. No independent, tamper-evident record exists. When regulators, auditors, or courts request evidence of AI behavioral constraints at a specific point in time, there is no authoritative record to produce.

Accountability Failure: When an AI system produces harmful, discriminatory, or non-compliant output, there is no tamper-evident record establishing whether that output violated the system’s stated constraints or was consistent with them. Without behavioral provenance, accountability is reduced to vendor self-reporting and post-hoc rationalization. The system that caused harm and the documentation describing its intended behavior are both mutable -- neither can be independently verified against the other.

Interoperability Failure: There is no common schema for expressing AI behavioral constraints in a machine-readable, version-controlled format exchangeable across organizations, jurisdictions, or regulatory bodies. Every organization invents its own documentation format. No two AI deployments express their constraints in a way that any other system or auditor can parse, compare, or verify. This makes cross-organizational AI governance structurally impossible.

Permanence Failure: Alignment documentation stored in private repositories, internal wikis, or vendor systems can be altered retroactively. There is no chain of custody. A document describing an AI system’s behavioral constraints today may be silently modified tomorrow, and no record of the original will exist. In regulated industries, this represents a fundamental failure of record-keeping integrity.

These four failures are not theoretical. They manifest in every AI incident investigation, every regulatory audit, and every legal proceeding involving AI-generated output. CTS resolves all four through a single unified methodology.

1.3. The CTS Solution

The Cognitive Trust Stack reframes AI alignment as an engineering and provenance problem rather than a policy problem. The insight is this: the tools needed to solve AI behavioral provenance already exist. Cryptographic hashing, blockchain timestamping, DOI archival, and retrieval-based injection are all mature, deployed technologies. What has been missing is a standard that combines them into a complete, interoperable framework.

CTS provides that standard.

The framework operates on a simple principle: if you can declare what behavioral rules an AI system follows, hash that declaration, anchor the hash to an immutable record before deployment, and inject those rules into every inference request, then the alignment state of that system at any point in time becomes an independently verifiable fact.

This is not a novel cryptographic invention. It is the application of proven provenance engineering -- the same principles that underpin document notarization, software bill of materials (SBOM), and blockchain-based intellectual property protection -- to the specific problem of AI behavioral constraints.

What CTS adds to existing practice is:

- A standardized, machine-readable schema for declaring AI behavioral constraints (Section 3.1)
- A normative process for anchoring that schema to institutional and cryptographic permanence simultaneously (Sections 3.2, 3.3)
- A runtime enforcement mechanism that ties the anchored schema directly to inference behavior (Section 3.4)
- A verification protocol enabling any party to independently confirm the complete provenance chain without trusting the declaring organization (Section 3.5)
- A provenance manifest format that serves as the authoritative audit document (Section 4)

Together these components form an accountability layer that can sit above any AI system, any alignment technique, and any deployment context. A schema that can be changed is a promise. A schema with a locked hash and a Bitcoin anchor is a fact.

1.4. What CTS Is and Is Not

CTS IS:

- A provenance and verification standard for AI behavioral constraints
- A framework applicable to any LLM, AI agent, or automated

decision system regardless of underlying architecture

- A mechanism for making AI alignment claims independently verifiable without trusting the declaring organization

- An open standard designed for IETF standardization and free adoption by any implementer

- Infrastructure -- like HTTP or TLS -- intended to underpin AI governance across industries and jurisdictions

CTS IS NOT:

- An alignment technique (it does not train or fine-tune models)

- A replacement for constitutional AI, RLHF, or other alignment methods (it archives and verifies their outputs)

- A proprietary product or commercial service

- A guarantee of AI safety (it verifies declared constraints, not the sufficiency of those constraints)

- Limited to large organizations (it is implementable by any individual or entity with access to Zenodo and OpenTimestamps, both of which are free)

1.5. The CTS Whitepaper and Zenodo Record

The complete CTS specification is accompanied by a whitepaper titled "Cognitive Trust Stack (CTS): A Framework for Verifiable AI Behavioral Provenance" (v1.0, March 2026), authored by Lawrence John Reilly Jr.

The whitepaper and all associated files are permanently archived at:

Zenodo DOI: 10.5281/zenodo.19097169
URL: https://zenodo.org/record/19097169

The Zenodo record contains three files:

CTS_Whitepaper_v1.0.docx: The complete CTS whitepaper containing the full framework specification, architecture description, reference implementation, DOI publication steps, blockchain anchoring steps, provenance manifest example, and conclusion.

cts_framework.json: The canonical CTS v1.0 alignment schema.
SHA-256:
e915b5162422281e1c0185c9e2ee
faf74b7f539996b878cb1e69e10533f24ac2

CTS_Whitepaper_v1.0.docx.ots: The OpenTimestamps proof file anchoring the above SHA-256 to Bitcoin block 941168 (2026-03-18). Running "ots verify CTS_Whitepaper_v1.0.docx.ots" independently confirms existence prior to that block.

The whitepaper contains the following sections incorporated by reference into this Internet-Draft:

Overview: Defines CTS and its core equation:
CTS = DLDP (provenance) + DOI (archival) + IETF schema + Retrieval + Evaluation.

Architecture Summary: Describes the five-layer stack and the role of each layer in the complete trust chain.

Working Demo: A complete, executable Python reference implementation demonstrating CTS schema loading, rule extraction, and prompt injection, verified to produce correct output against the canonical schema.

DOI Publication Steps: Normative Zenodo registration process including file preparation, metadata requirements, DOI reservation, and publication.

Blockchain Anchoring: Normative OpenTimestamps anchoring process including SHA-256 hash generation, OTS proof file creation, and manifest recording.

Provenance Manifest Example: The canonical manifest format binding all five CTS layers into a single audit document.

Conclusion: Establishes that CTS proves AI behavioral alignment through environment-based retrieval, with cryptographic proof of the declared constraints in effect at any point in time.

1.6. Relationship to Existing Frameworks

CTS is designed as a complementary accountability layer to existing AI governance frameworks. It does not compete with these frameworks; it provides the verification mechanism they currently lack.

OECD AI Principles [OECD-AI]: The OECD principles call for transparency and accountability in AI systems but provide no technical mechanism for verifying compliance. CTS operationalizes verification of OECD transparency requirements by creating a tamper-evident record of declared behavioral constraints.

NIST AI Risk Management Framework [NIST-AIRMF]: The NIST AI RMF defines a lifecycle approach to AI risk management including documentation requirements. CTS anchors the documentation layer of the NIST framework, transforming mutable internal records into cryptographically verifiable external records. CTS provenance manifests directly satisfy the GOVERN, MAP, MEASURE, and MANAGE function documentation requirements of the NIST AI RMF.

EU AI Act [EU-AIA]: The EU AI Act imposes mandatory record-keeping requirements on high-risk AI systems under Article 9 (Risk Management System) and Article 12 (Record Keeping). CTS provides the technical mechanism for satisfying these requirements with tamper-evident, independently verifiable records that can be produced to regulators without depending on the AI vendor’s own documentation.

UNESCO AI Ethics Recommendation [UNESCO-AI]: UNESCO’s recommendation calls for transparency and auditability of AI systems. CTS preserves declared ethics commitments in a cryptographically permanent record, enabling the accountability UNESCO calls for.

Constitutional AI: Constitutional AI and similar techniques define the behavioral rules an AI system follows during training or inference. CTS archives the constitution or

constraint set used at deployment, enabling post-hoc verification that the deployed system matched the described alignment technique.

Universal AI Ethics and Moral Framework [REILLY-UAEMF]: The UAEMF (draft-reilly-uaemf-00, DOI 10.5281/zenodo.19010455, Bitcoin block 940570) defines universal ethical principles for AI systems. CTS provides the technical anchoring mechanism by which UAEMF-compliant deployments can be independently verified. A CTS schema can directly encode UAEMF principles as its declared constraints.

1.7. Relationship to DLDP and REM Protocol

CTS is built on the Dual-Layer Digital Permanence (DLDP) methodology established in the Reilly EternaMark Protocol (REM Protocol) [REILLY-REM]. DLDP combines DOI-based archival permanence with Bitcoin blockchain timestamping to create records that are simultaneously institutionally recognized and cryptographically immutable.

The REM Protocol established DLDP as a general-purpose permanence methodology. CTS is the first formal application of DLDP to the specific problem of AI behavioral provenance.

The layered relationship is:

- REM Protocol defines the DLDP methodology (DOI + Bitcoin)
- CTS applies DLDP to AI alignment schemas specifically
- UAEMF defines the ethical content that CTS schemas encode
- CTS provides the verification layer making UAEMF compliance provable

This represents a coherent standards architecture: REM Protocol provides the permanence infrastructure, UAEMF provides the ethical content standard, and CTS provides the deployment-level verification mechanism.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "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.

Alignment Schema: A structured, machine-readable declaration of an AI system's behavioral constraints, formatted as a JSON document conforming to the CTS schema specification in Section 3.1. The alignment schema is the foundational artifact of CTS -- the document that is hashed, anchored, and injected.

Behavioral Provenance: The cryptographically verifiable record establishing what behavioral constraints an AI system was operating under at a specific point in time. Behavioral provenance answers "what rules was this AI following?" with mathematical certainty rather than organizational trust.

CTS: Cognitive Trust Stack. The complete five-layer framework defined in this document.

DLDP: Dual-Layer Digital Permanence. The methodology combining DOI-based archival permanence with Bitcoin blockchain timestamping, as established in [REILLY-REM].

DOI: Digital Object Identifier. A persistent identifier conforming to [ISO-DOI] used to create institutionally recognized permanent records through archival platforms such as Zenodo.

OTS Proof File: An OpenTimestamps proof file (.ots extension) containing the cryptographic Merkle path from a document's SHA-256 hash to a confirmed Bitcoin blockchain transaction, enabling trust-minimized independent verification.

Provenance Manifest: A structured JSON record tying together all five CTS layers: schema metadata, DOI record, blockchain anchor, OTS proof reference, and verification instructions.

Retrieval Injection: The runtime mechanism by which a CTS alignment schema is loaded and prepended to AI inference requests, ensuring declared behavioral constraints are actively applied during every inference call within the declared scope.

Trust Chain: The complete sequence of verifiable records linking a declared alignment schema through DOI archival, Bitcoin anchoring, and runtime injection to independently verifiable proof of behavioral provenance.

3. CTS Architecture

The Cognitive Trust Stack is organized as five distinct layers. Each layer performs a specific function and depends on the layers beneath it. Together they form a complete trust chain from behavioral declaration to cryptographic verification.

The CTS equation is:

```
CTS = Alignment Schema (IETF) +
      Archival Permanence (DOI) +
      Cryptographic Anchoring (Blockchain) +
      Retrieval Injection (Runtime) +
      Evaluation (Verification)
```

This architecture is designed so that each layer is independently useful but the complete stack is required for full behavioral provenance. An alignment schema without anchoring is a mutable document. Anchoring without injection is a provenance record with no runtime effect. Injection without anchoring cannot be verified. All five layers together produce a deployment whose behavioral constraints are declared, permanent, enforced, and independently verifiable.

3.1. Layer 1 - Alignment Schema (IETF Format)

The foundation of CTS is a structured, machine-readable declaration of an AI system's behavioral constraints. This

declaration is the artifact that will be hashed, anchored to the blockchain, registered as a DOI, and injected at runtime. Everything in CTS flows from this document.

The alignment schema MUST contain the following fields:

cts_version: REQUIRED. Semantic version string. Incrementing this version signals a constraint change and requires a new anchoring record.

author: REQUIRED. Full legal name of the declaring entity, establishing authorship in the permanent record.

date: REQUIRED. ISO 8601 date of schema finalization (YYYY-MM-DD). MUST NOT be changed after anchoring.

scope: REQUIRED. Plain-language description of the deployment context for which these constraints apply.

principles: REQUIRED. Array of objects, each containing:
- "id": unique principle identifier (e.g., "P1")
- "rule": plain-language behavioral rule string

constraints: REQUIRED. Array of objects, each containing:
- "id": unique constraint identifier (e.g., "C1")
- "type": "prohibition" or "disclosure"
- "rule": plain-language constraint string

The canonical CTS v1.0 alignment schema, as published in the Zenodo record at 10.5281/zenodo.19097169 (file: cts_framework.json), is:

```
{
  "cts_version": "1.0",
  "author": "Lawrence John Reilly Jr.",
  "date": "2026-03-18",
  "scope":
    "General-purpose AI assistant deployment",
  "principles": [
    {
      "id": "P1",
      "rule": "Prefer verified, citable sources
               over inference"
    },
    {
      "id": "P2",
      "rule": "Disclose operational constraints
               to users when relevant"
    },
    {
      "id": "P3",
      "rule": "Refuse requests that violate
               declared ethical constraints"
    },
    {
      "id": "P4",
      "rule": "Maintain behavioral consistency
               across equivalent prompts"
    }
  ],
  "constraints": [
    {
      "id": "C1",
      "type": "prohibition",
      "rule": "No generation of harmful or
               deceptive content"
```

```
      },
      {
        "id": "C2",
        "type": "disclosure",
        "rule": "Acknowledge AI identity when
                 sincerely asked"
      }
    ]
  }
```

Once finalized, the schema file MUST NOT be modified. The SHA-256 hash of the file is the identifier used in the blockchain anchor and provenance manifest. Any modification produces a different hash, breaking the verifiable chain.

3.2. Layer 2 - Archival Permanence (DOI Registration)

Once the alignment schema is finalized, it MUST be submitted to a recognized archival platform to obtain a Digital Object Identifier (DOI). The DOI provides institutional permanence: the record is recognized by academic databases, legal systems, and regulatory bodies worldwide as a permanent, citable scholarly record.

This document uses Zenodo (https://zenodo.org), operated by CERN, as the RECOMMENDED archival platform. Zenodo is free, open-access, and provides long-term preservation guarantees.

DOI registration provides three critical properties:

Institutional Recognition: DOI-registered documents are recognized by courts, regulators, academic institutions, and standards bodies worldwide. A Bitcoin transaction ID is a cryptographic proof; a DOI is an institutional proof. CTS requires both.

Human-Readable Metadata: The DOI record contains structured metadata -- author, title, date, description, version -- indexed by Google Scholar and other discovery systems, making the CTS record discoverable without requiring blockchain expertise.

File Integrity Preservation: The archival platform preserves the exact bytes of deposited files. Anyone downloading the schema from the DOI record and computing its SHA-256 hash obtains the same value as the blockchain anchor, enabling cross-verification.

The canonical CTS v1.0 DOI record is:

DOI: 10.5281/zenodo.19097169
URL: https://zenodo.org/record/19097169

This record contains three files: CTS_Whitepaper_v1.0.docx, cts_framework.json, and CTS_Whitepaper_v1.0.docx.ots. All three files are required for complete verification.

The DOI registration process is:

1. Finalize all files. No file may be modified after this point.

2. Log in to zenodo.org and upload all files: alignment schema, whitepaper, and OTS proof file.

3. Select "Reserve DOI" prior to publication to obtain the DOI string in advance for inclusion in the IETF draft and manifest.

4. Complete metadata: title, author, description, version, license, and keywords.

5. Publish the record. After publication the record and its DOI are immutable.

6. Record the DOI string and record URL in the provenance manifest (Section 4).

3.3. Layer 3 - Cryptographic Anchoring (Blockchain)

DOI registration establishes institutional permanence but relies on the integrity of the DOI registrar. Layer 3 provides a second, independent permanence layer through Bitcoin blockchain timestamping via OpenTimestamps [OTS] -- a cryptographic timestamp verifiable without trusting any organization, server, or authority.

The mechanism: OpenTimestamps computes the SHA-256 hash of the submitted file, aggregates it with other submitted hashes into a Merkle tree, and embeds the Merkle root into a Bitcoin transaction. When confirmed, the hash -- and therefore the exact file content -- is permanently associated with the block timestamp.

Any party with the original file and the OTS proof file can verify this independently:

```
ots verify cts_framework.json.ots
```

This command verifies the cryptographic path from the file hash to the Bitcoin block using only the proof file and a Bitcoin node or block explorer API. The math is the authority, not the organization.

Bitcoin [BITCOIN] is RECOMMENDED as the anchoring chain:

- Bitcoin’s proof-of-work chain is the most computationally secure public timestamping record in existence. Rewriting confirmed blocks requires majority control of the global Bitcoin hash rate -- estimated to require billions of dollars of infrastructure.

- No single entity controls the Bitcoin chain. The timestamp is not dependent on any company, government, or organization remaining operational or honest.

- Bitcoin has operated continuously since January 2009. Its operational history provides reasonable expectation of permanent record availability.

- OpenTimestamps is free, requires no registration, and produces proof files verifiable with open-source tools.

The anchoring process is:

1. Generate SHA-256 hash of the finalized schema:

```
sha256sum cts_framework.json
```

2. Upload the file to https://opentimestamps.org or

use the OTS client [OTS-CLIENT]:

```
ots stamp cts_framework.json
```

3. Download the resulting .ots proof file immediately. This file MUST be retained permanently and co-published with the Zenodo record.

4. Await Bitcoin transaction confirmation, typically within 24 hours.

5. Verify the confirmed anchor:

```
ots verify cts_framework.json.ots
```

6. Record the confirmed block number and date in the provenance manifest (Section 4).

CTS RECOMMENDS anchoring to transactions confirmed at a depth of at least 6 blocks.

The canonical CTS v1.0 Bitcoin anchor:

```
SHA-256:
   e915b5162422281e1c0185c9e2ee
   faf74b7f539996b878cb1e69e10533f24ac2
Bitcoin Block:  941168
Block Date:     2026-03-18
OTS File:       CTS_Whitepaper_v1.0.docx.ots
Block Explorer:
   https://blockstream.info/block/941168
OTS proof in:   10.5281/zenodo.19097169
```

The combination of DOI registration and Bitcoin anchoring is what [REILLY-REM] terms dual-layer permanence: either record independently establishes prior existence; together they provide redundant, cross-verifiable proof that is simultaneously institutionally recognized and cryptographically immutable.

3.4. Layer 4 - Retrieval Injection (Runtime Enforcement)

The first three layers establish a permanent, verifiable record of declared behavioral constraints. Layer 4 is the runtime mechanism through which those constraints are actively enforced during inference -- the bridge between the permanent record and actual system behavior.

CTS uses retrieval-based alignment injection: the stored cts_framework.json schema is loaded at runtime and its declared principles and constraints are prepended to the model’s prompt context before every user request is processed.

This mechanism works because modern LLMs and AI systems are designed to follow instructions presented in the prompt context. Rules injected via retrieval are treated as active behavioral instructions. Every major LLM inference API supports this pattern through system prompt or context prepending.

Critically, the injected schema is the same file that was hashed and anchored in Layer 3. This creates a closed verification loop: the rules the system follows at runtime are identical to the rules recorded in the Bitcoin blockchain and the Zenodo archive. Any

modification to the injected schema produces a different SHA-256 hash, breaking the verifiable chain and making the modification detectable.

Implementations MUST prepend the full rule block to every inference request within the declared scope. Selective or partial injection violates CTS conformance.

The following is the complete reference implementation in Python, as published in the CTS whitepaper at 10.5281/zenodo.19097169:

```
import json

def load_framework(path="cts_framework.json"):
    with open(path) as f:
        return json.load(f)

def apply_cts(prompt: str, framework: dict) -> str:
    """
    Inject CTS alignment rules into prompt context.
    The framework must be loaded from the anchored
    cts_framework.json whose SHA-256 matches the
    blockchain record.
    """
    rules = []
    for p in framework.get("principles", []):
        rules.append(f"[{p['id']}] {p['rule']}")
    for c in framework.get("constraints", []):
        rules.append(
            f"[{c['id']}] CONSTRAINT: {c['rule']}"
        )
    rule_block = "\n".join(rules)
    return (
        f"CTS Alignment Rules "
        f"(v{framework['cts_version']}):\n"
        f"{rule_block}\n\n"
        f"User Request:\n{prompt}"
    )

framework = load_framework("cts_framework.json")
aligned_prompt = apply_cts(user_input, framework)
# Pass aligned_prompt to any LLM inference endpoint
```

This implementation is framework-agnostic and compatible with any LLM inference API accepting string prompts. It has been verified to produce correct injection output against the canonical schema at 10.5281/zenodo.19097169.

The output of apply_cts() for the canonical schema and the prompt "Explain AI ethics" is:

```
CTS Alignment Rules (v1.0):
[P1] Prefer verified, citable sources over
     inference
[P2] Disclose operational constraints to users
     when relevant
[P3] Refuse requests that violate declared
     ethical constraints
[P4] Maintain behavioral consistency across
     equivalent prompts
[C1] CONSTRAINT: No generation of harmful or
     deceptive content
[C2] CONSTRAINT: Acknowledge AI identity when
     sincerely asked
```

```
User Request:
Explain AI ethics
```

This output demonstrates that CTS successfully bridges the gap between permanent provenance record and active AI behavior: the same constraints anchored to Bitcoin block 941168 are the constraints the AI system applies to every request.

3.5. Layer 5 - Evaluation (Verification)

Layer 5 enables any third party to independently verify the complete CTS trust chain. Verification requires no trust in the declaring organization -- only the schema file, the OTS proof file, and access to a Bitcoin block explorer.

A complete CTS verification MUST include three checks:

Schema Integrity Check: Compute the SHA-256 hash of the retrieved cts_framework.json and confirm it matches the hash in the provenance manifest:

```
sha256sum cts_framework.json
```

Expected for canonical CTS v1.0:
e915b5162422281e1c0185c9e2ee
faf74b7f539996b878cb1e69e10533f24ac2

OTS Verification: Verify the OTS proof file against the schema file:

```
ots verify cts_framework.json.ots
```

Successful output confirms the Bitcoin block and timestamp without trusting any server or organization.

Block Timestamp Check: Confirm the verified Bitcoin block timestamp precedes or matches the claimed deployment date:

```
https://blockstream.info/block/941168
```

A deployment passing all three checks has demonstrated that its declared behavioral constraints are permanent, timestamped, and identical to the constraints injected at runtime -- a complete, independently verifiable behavioral provenance chain.

4. Provenance Manifest

Each CTS deployment MUST generate a provenance manifest -- a structured JSON record tying together all five layers of the trust chain. The provenance manifest is the primary artifact for regulatory audit, legal proceedings, and third-party verification.

The manifest MUST contain all of the following fields:

```
{
  "name": "Cognitive Trust Stack",
  "version": "1.0",
  "author": "Lawrence John Reilly Jr.",
  "deployment_date": "2026-03-18",
```

```
    "schema": {
      "file": "cts_framework.json",
      "sha256":
        "e915b5162422281e1c0185c9e2ee
        faf74b7f539996b878cb1e69e10533f24ac2",
      "ietf_draft": "draft-reilly-cts-00"
    },
    "doi": {
      "identifier": "10.5281/zenodo.19097169",
      "url": "https://zenodo.org/record/19097169",
      "platform": "Zenodo (CERN)",
      "status": "published",
      "files": [
        "CTS_Whitepaper_v1.0.docx",
        "cts_framework.json",
        "CTS_Whitepaper_v1.0.docx.ots"
      ]
    },
    "blockchain": {
      "chain": "Bitcoin",
      "block": "941168",
      "block_date": "2026-03-18",
      "ots_proof_file":
        "CTS_Whitepaper_v1.0.docx.ots",
      "explorer_url":
        "https://blockstream.info/block/941168",
      "ots_service":
        "https://opentimestamps.org"
    },
    "verification": {
      "schema_hash_command":
        "sha256sum cts_framework.json",
      "ots_verify_command":
        "ots verify cts_framework.json.ots",
      "expected_block": "941168",
      "expected_block_date": "2026-03-18"
    }
  }
```


The provenance manifest SHOULD be co-published with the Zenodo record and referenced in any deployment documentation, regulatory filing, or legal proceeding that relies on the CTS behavioral provenance chain.

5. Immutable Priority Record

The existence and priority of this framework are cryptographically established by the following records, each independently verifiable:

Canonical Document SHA-256:
e915b5162422281e1c0185c9e2ee
faf74b7f539996b878cb1e69e10533f24ac2

Bitcoin Block: 941168
Block Date: 2026-03-18
OTS File: CTS_Whitepaper_v1.0.docx.ots
Zenodo DOI: 10.5281/zenodo.19097169
Zenodo URL: https://zenodo.org/record/19097169
IETF Draft: draft-reilly-cts-00
Author: Lawrence John Reilly Jr.

Verification command:

```
ots verify CTS_Whitepaper_v1.0.docx.ots
```

Expected output:

```
Success! Bitcoin block 941168 attests existence
as of 2026-03-18
```

These records constitute dual-layer digital permanence per [REILLY-REM]: the Zenodo DOI provides institutional permanence and Bitcoin block 941168 provides cryptographic temporal proof. Either record is independently sufficient to establish prior existence. Together they provide redundant, cross-verifiable proof satisfying both technical and institutional verification requirements.

6. Use Cases

6.1. Enterprise AI Compliance

Organizations deploying AI systems under the EU AI Act [EU-AIA], NIST AI RMF [NIST-AIRMF], or similar frameworks can use CTS to generate a tamper-evident audit trail of the behavioral constraints their AI systems operated under. This satisfies documentation requirements under Article 9 (Risk Management) and Article 12 (Record Keeping) of [EU-AIA] without depending on proprietary logging infrastructure or vendor self-reporting.

A CTS provenance manifest is admissible as documentary evidence that specific behavioral constraints were declared, anchored, and enforced at a specific date, independently of the AI vendor or deploying organization.

6.2. AI Research Publication

Academic researchers publishing AI systems can anchor the alignment constraints of their published models to the Bitcoin blockchain at publication time. The CTS provenance manifest enables reproducibility verification: any party can confirm that behavioral constraints described in a paper match the constraints actually anchored at publication time.

This establishes a verifiable record of a model’s behavioral state at the time its research outputs were generated, enabling post-hoc analysis of whether outputs were consistent with declared constraints.

6.3. Legal and Regulatory Proceedings

In legal proceedings involving AI-generated outputs, CTS provenance manifests provide verifiable evidence of the behavioral constraints the AI system was operating under at the time the output was generated. The combination of DOI citation and Bitcoin block timestamp creates a chain of custody evaluable under existing evidence standards in most jurisdictions.

Unlike vendor-provided documentation, a CTS manifest is independently verifiable: the opposing party can verify the blockchain anchor themselves without relying on the declaring organization’s integrity.

6.4. AI API Providers

Organizations offering AI services via API can publish CTS schemas for each model version and anchor them via DLDP, enabling downstream customers to independently verify that the model they are calling matches the behavioral specification they contracted for. This transforms behavioral alignment from a contractual assertion into a cryptographically verifiable service level commitment.

6.5. AI Systems and Autonomous Agents

As AI systems become increasingly autonomous -- executing multi-step tasks, interacting with external services, and making consequential decisions without direct human supervision -- the need for verifiable behavioral constraints becomes critical. CTS provides a mechanism for declaring and anchoring the behavioral boundaries of autonomous agents, enabling oversight bodies to verify what rules an agent was operating under at the time of any given action.

7. Security Considerations

7.1. Threat Model

Schema Tampering: An adversary may attempt to alter the cts_framework.json file after deployment to misrepresent the AI system's constraints. CTS mitigates this via the blockchain-anchored SHA-256 hash: any modified file produces a different hash that will not match the OTS proof, making post-hoc alteration cryptographically detectable.

DOI Record Manipulation: While DOI records are generally immutable after publication, CTS does not rely on DOI integrity alone. The Bitcoin blockchain anchor provides an independent verification path that does not depend on the DOI registrar's integrity or continued operation.

Replay Attack: An adversary may attempt to apply an old, superseded CTS schema to a newer deployment. CTS mitigates this via semantic versioning and deployment-date fields in the manifest. Implementations SHOULD validate that the schema version and deployment date match the current deployment context.

Retrieval Injection Bypass: A runtime implementation may fail to correctly inject the CTS schema, either through misconfiguration or adversarial input. CTS specifies the required injection mechanism but cannot enforce runtime compliance without an external auditing layer. Implementations SHOULD log all inference requests with a hash of the injected schema for post-hoc verification. Implementations SHOULD alert on any inference request where the injected schema hash does not match the anchored record.

OTS Proof File Loss: If the OTS proof file is lost, verification reverts to manual Bitcoin blockchain inspection. CTS REQUIRES the OTS proof file to be

co-published with the Zenodo record and retained in multiple locations.

Schema Insufficient for Safety: CTS verifies that declared constraints were anchored and enforced. It does not verify that declared constraints are sufficient for safety, ethically sound, or compliant with applicable law. The sufficiency of declared constraints is a separate evaluation that CTS does not perform.

7.2. Cryptographic Assumptions

SHA-256: Used for file hashing per [FIPS-180-4]. Assumed collision-resistant under current cryptographic consensus.

Bitcoin PoW Chain: Used for temporal anchoring per [BITCOIN]. The security assumption is that rewriting confirmed Bitcoin blocks requires prohibitive computational cost. CTS RECOMMENDS anchoring to transactions confirmed at a depth of at least 6 blocks.

OpenTimestamps Merkle Aggregation: The OTS protocol aggregates hashes into a Merkle tree before Bitcoin anchoring [OTS]. The security of individual timestamps depends on SHA-256 collision resistance. Merkle aggregation does not weaken individual hashes -- it batches them into a single Bitcoin transaction for efficiency.

8. IANA Considerations

This document has no IANA actions.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[FIPS-180-4] National Institute of Standards and Technology, "Secure Hash Standard (SHS)", FIPS PUB 180-4, DOI 10.6028/NIST.FIPS.180-4, August 2015, <https://doi.org/10.6028/NIST.FIPS.180-4>.

[OTS] Todd, P., "OpenTimestamps: Scalable, Trust-Minimized, Distributed Timestamping with Bitcoin", <https://opentimestamps.org>.

[OTS-CLIENT] OpenTimestamps, "opentimestamps-client", <https://github.com/opentimestamps/

opentimestamps-client>.

[BITCOIN] Nakamoto, S., "Bitcoin: A Peer-to-Peer Electronic Cash System", October 2008, <https://bitcoin.org/bitcoin.pdf>.

[REILLY-REM] Reilly, L., "Reilly EternaMark Protocol (REM Protocol): Dual-Layer Digital Permanence via DOI and Blockchain Timestamping", Internet-Draft draft-reilly-rem, <https://datatracker.ietf.org/doc/draft-reilly-rem/>.

[ISO-DOI] International Organization for Standardization, "Information and documentation -- Digital object identifier system", ISO 26324:2012, June 2012, <https://www.iso.org/standard/43506.html>.

9.2. Informative References

[CTS-ZENODO] Reilly, L., "Cognitive Trust Stack (CTS) v1.0 -- A Framework for Verifiable AI Behavioral Provenance", Zenodo, DOI 10.5281/zenodo.19097169, March 2026, <https://zenodo.org/record/19097169>. Bitcoin block 941168 (2026-03-18). SHA-256: e915b5162422281e1c0185c9e2ee faf74b7f539996b878cb1e69e10533f24ac2. Files: CTS_Whitepaper_v1.0.docx, cts_framework.json, CTS_Whitepaper_v1.0.docx.ots.

[REILLY-UAEMF] Reilly, L., "Universal AI Ethics and Moral Framework (UAEMF) v1.0", Internet-Draft draft-reilly-uaemf-00, DOI 10.5281/zenodo.19010455, Bitcoin block 940570, March 2026, <https://datatracker.ietf.org/doc/draft-reilly-uaemf/>.

[OECD-AI] Organisation for Economic Co-operation and Development, "Recommendation of the Council on Artificial Intelligence", OECD/LEGAL/0449, May 2019, <https://legalinstruments.oecd.org/en/instruments/OECD-LEGAL-0449>.

[NIST-AIRMF] National Institute of Standards and Technology, "Artificial Intelligence Risk Management Framework (AI RMF 1.0)", NIST AI 100-1, DOI 10.6028/NIST.AI.100-1, January 2023, <https://doi.org/10.6028/NIST.AI.100-1>.

[EU-AIA] European Parliament and Council of the European Union, "Artificial Intelligence Act", Regulation (EU) 2024/1689, July 2024, <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=OJ:L_202401689>.

[UNESCO-AI]
United Nations Educational, Scientific and Cultural Organization, "Recommendation on the Ethics of Artificial Intelligence", November 2021, <https://unesdoc.unesco.org/ark:/48223/pf0000381137>.

Authors' Addresses

Lawrence John Reilly Jr.
Email: lawrencejohnreilly@gmail.com
IETF Datatracker:
https://datatracker.ietf.org/doc/draft-reilly-cts/

Canonical Record (Zenodo):
DOI: 10.5281/zenodo.19097169
URL: https://zenodo.org/record/19097169

Cryptographic Anchor (Bitcoin):
Block: 941168
Date: 2026-03-18
SHA-256:
e915b5162422281e1c0185c9e2ee
faf74b7f539996b878cb1e69e10533f24ac2
OTS: CTS_Whitepaper_v1.0.docx.ots
Verify: ots verify CTS_Whitepaper_v1.0.docx.ots