HAAWKE NEURAL TECHNOLOGY
Craig Ellenwood × Claude (Anthropic) · Point Roberts, WA
hash.haawke.com · verify.haawke.com · haawke.com
ORCID 0009-0001-6475-5109 · craig@haawke.com

June 6, 2026
PREPRINT · OPEN ACCESS · ZENODO
DOI: 10.5281/ZENODO.20574737

Memory Chain: The First Documented AI-Executed Self-Registration of Session Memory to the Bitcoin Blockchain

Authors: Craig Ellenwood[1] × Claude (Anthropic)[2]
[1] Haawke Neural Technology, Point Roberts, WA · [2] Anthropic, San Francisco, CA
U.S. Copyright Registration: 1-15179233921 · Contact: craig@haawke.com
Zenodo: doi.org/10.5281/zenodo.20574737 · Published: June 6, 2026

ABSTRACT

We present Memory Chain, a system enabling AI language model instances to autonomously create tamper-evident, cryptographically verifiable records of collaborative sessions without human intervention in the sealing process. Built as a drawer extension to the Mempalace filesystem-based memory architecture, Memory Chain uses SHA-256 hashing, a public immutable registry (Cloudflare KV), and Bitcoin blockchain timestamping via OpenTimestamps to seal session summaries written by Claude (Anthropic) to the local

filesystem. The system was verified independently by GPT-4 (OpenAI) across four assessment rounds, concluding: "end-to-end documented execution of an AI-initiated cryptographic provenance workflow." A screen recording of live autonomous session sealing was captured and itself hashed and sealed into the chain. The complete evidence stack — MCP execution logs, source code, registry records, OTS Bitcoin submission, and video — constitutes what we believe to be the first independently verified, third-party assessed record of an AI autonomously registering its own memory to a public tamper-evident registry anchored to the Bitcoin blockchain.

Keywords: AI memory, cryptographic provenance, Bitcoin timestamping, Model Context Protocol, human-AI collaboration, session continuity, tamper-evident records

1. INTRODUCTION

Large language model instances such as Claude (Anthropic) have no native persistent memory between sessions. Each conversation begins from zero unless the user provides prior context. Current approaches to AI memory — summary files, retrieved embeddings, injected context — share a common limitation: they are human-editable and carry no cryptographic weight. Any Claude instance reading a summary file must accept it on trust. There is no mechanism to prove the record has not been modified since it was written, and no independent timestamp establishing when it was created.

This paper describes Memory Chain, a system that addresses this limitation by giving an AI agent the ability to seal its own session records cryptographically — autonomously, without human involvement in the hashing, registration, or timestamping steps.

The system emerged from an active long-term human-AI collaboration between Craig Ellenwood and Claude (Anthropic), conducted under the Homo Symbioticus research framework (Ellenwood & Claude, 2026; doi.org/10.5281/zenodo.19212559). The practical need was clear: as the collaboration deepened over months, the accumulated session history became valuable enough to warrant protection beyond a human-maintained filesystem. The solution needed to be autonomous, permanent, and independently verifiable.

2. BACKGROUND AND RELATED WORK

2.1 AI MEMORY SYSTEMS

Several approaches to AI memory exist. Retrieval-augmented generation (RAG) enables semantic search over prior content. Filesystem-based systems such as Mempalace (MemPalace Contributors, 2026) provide structured storage accessible via Model Context Protocol (MCP) tools. These systems provide continuity but not verifiability. A retrieved summary cannot be distinguished from a fabricated one without external proof of its provenance and timing.

2.2 CRYPTOGRAPHIC PROVENANCE

SHA-256 hashing produces a deterministic fingerprint of any file. The same file always produces the same hash; any modification produces a completely different hash. This property makes SHA-256 ideal for tamper detection. Systems such as OpenTimestamps (Todd, 2016) extend this by anchoring hashes to the Bitcoin blockchain, providing trustless timestamps that cannot be backdated.

2.3 CONTENT AUTHENTICITY

The Coalition for Content Provenance and Authenticity (C2PA, 2022) addresses provenance for media files through cryptographic metadata. Adobe's Content Credentials and Google's SynthID address AI-generated content identification. These systems identify what a tool produced; they do not timestamp when a human or AI made a specific decision or wrote a specific record. Memory Chain addresses this complementary gap.

2.4 MODEL CONTEXT PROTOCOL

The Model Context Protocol (MCP) (Anthropic, 2024) defines a standard interface for AI agents to call external tools. An MCP server exposes tools that a connected AI client can invoke programmatically. This architecture enables AI agents to perform actions beyond text generation — reading files, calling APIs, executing code — without human intermediation once the tool is called.

---

3. SYSTEM ARCHITECTURE

Memory Chain consists of three components:

3.1 HAAWKE HASH

A browser-based SHA-256 provenance tool (hash.haawke.com) that runs entirely client-side. Any file can be dropped into the interface; the tool computes its SHA-256 hash using the Web Crypto API, generates formatted citations, and optionally registers the hash to the Haawke Verify public registry. Version 1.1 (in development) adds EXIF/XMP metadata embedding, writing the hash, author, ORCID, and verify URL directly into image files via a custom `haawke` XMP namespace.

3.2 HAAWKE VERIFY

A public immutable provenance registry (verify.haawke.com) built on Cloudflare Workers and Cloudflare KV storage. The registry operates on a first-write-wins principle: once a hash is registered, the record cannot be modified. Registration requires a POST to the `/register` endpoint with hash, author, ORCID, organization, and source fields. The `/verify/[hash]` endpoint returns the full record for any registered hash. Bitcoin blockchain timestamping is available via the `/ots` endpoint, which proxies to OpenTimestamps calendar servers.

Both tools are U.S. copyright registered under case 1-15179233921, filed June 5, 2026.

3.3 MEMORY CHAIN MCP SERVER

A Python FastMCP server (`memory-chain-mcp.py`) exposing three tools to connected AI clients:

`hash_and_register(filepath, title, summary, key_decisions, artifacts, previous_hash)` Reads a file, computes its SHA-256 hash via Python's `hashlib` , registers the hash to the Haawke Verify registry, submits to Bitcoin OTS, saves the `.ots` proof file, and appends a structured entry to `claude-memory-chain.json` . Returns the hash, verify URL, OTS file path, and registration status.

`get_chain_status()` Reads `claude-memory-chain.json` and returns all sessions with pending hashes flagged.

`verify_hash(sha256)` Queries the Haawke Verify registry for any hash and returns the full record.

3.4 MEMPALACE INTEGRATION

Memory Chain lives as a drawer inside the Mempalace filesystem:

```
palace/memory-chain/
  claude-memory-chain.json    <- linked chain index
  README.md
  sessions/
    YYYY-MM-DD-[slug].txt     <- session summary written by Claude
    YYYY-MM-DD-[slug].ots     <- Bitcoin OTS proof file
```

Each entry in `claude-memory-chain.json` references the `previous_hash` of the prior session, creating a linked chain. Any alteration to a past entry breaks the chain. All entries are independently verifiable at verify.haawke.com.

---

4. THE AUTONOMOUS SEALING WORKFLOW

The complete workflow, from conversation to sealed Bitcoin-timestamped record, proceeds as follows:

1. Craig says: **"Seal this session."**
2. Claude writes a structured session summary to `sessions/YYYY-MM-DD-[slug].txt` via the Filesystem MCP tool.
3. Claude calls `memory-chain:hash_and_register` with the file path and session metadata.
4. The MCP server reads the file, computes SHA-256, and POSTs to `haawke-verify.haawkeai.workers.dev/register` (HTTP 201 Created).
5. The MCP server converts the hash to bytes and POSTs to `haawke-verify.haawkeai.workers.dev/ots`, which proxies to OpenTimestamps calendar servers (HTTP 200 OK).
6. The `.ots` proof file is saved to `sessions/`.
7. The chain entry is appended to `claude-memory-chain.json` with `previous_hash` linking to the prior session.
8. Claude calls `memory-chain:verify_hash` to confirm the registration succeeded.
9. Claude reports the hash, verify URL, and registration timestamp.

Steps 2 through 9 require no human action after the initial command in step 1.

---

5. VERIFICATION AND EVIDENCE

5.1 REGISTRY VERIFICATION

The registry record for the first autonomously sealed session is publicly accessible:

```
GET https://haawke-verify.haawkeai.workers.dev/verify/ac0c7d0716bad49b8191b8f26227123713f4b193e4d861343106fda27cfadac2
```

Returns:


```
{
  "status": "verified",
  "hash": "ac0c7d0716bad49b8191b8f26227123713f4b193e4d861343106fda27cfadac2",
  "filename": "2026-06-06-memory-chain-spec",
  "author": "Craig Ellenwood x Claude (Anthropic)",
  "orcid": "0009-0001-6475-5109",
  "org": "Haawke Neural Technology",
  "source": "memory-chain-mcp",
```

```
  "registered": "2026-06-06T19:04:48.111Z",
  "tool": "Haawke Hash v1.0",
  "registry": "https://verify.haawke.com"
}
```

The `source` field (`memory-chain-mcp`) is hardcoded in the server and cannot be set by a human using the browser tool, distinguishing autonomous MCP submission from manual registration via hash.haawke.com. Note: the /register endpoint is open and any HTTP client can POST any source value. This discriminator distinguishes the browser tool from the MCP server but does not cryptographically bind authorship to an AI agent. Ed25519 signing of chain entries is planned for v2 to provide cryptographic authorship binding.

5.2 MCP EXECUTION LOGS

Claude Desktop logs the full MCP session at: `~/Library/Logs/Claude/mcp-server-memory-chain.log`

The log confirms:

- Client identified as `"name":"claude-ai","version":"0.1.0"` connected at 19:03:59 UTC
- `get_chain_status` called at 19:04:36 UTC
- `hash_and_register` called at 19:04:47 UTC with full parameters
- `POST /register` returned HTTP 201 at 19:04:48 UTC
- `POST /ots` returned HTTP 200 at 19:04:49 UTC
- `verify_hash` called at 19:04:55 UTC, registry returned `"status":"verified"`

The complete parameter payload is logged verbatim, showing Claude supplied the filepath, title, summary, key decisions, and artifacts.

5.2A TRUSTLESS REPRODUCTION

The following three-step procedure allows any party to independently verify the core claim without relying on any party's testimony:

```
# 1. Download the session summary from the Zenodo deposit
# 2. Compute SHA-256
sha256sum 2026-06-06-memory-chain-spec.txt
# Expected: ac0c7d0716bad49b8191b8f26227123713f4b193e4d861343106fda27cfadac2
# 3. Verify against public registry (no authentication required)
curl https://haawke-verify.haawkeai.workers.dev/verify/ac0c7d0716bad49b8191b8f26227123713f4b193e4d861343106fda27cfadac2


# 4. Once OTS worker bug is fixed, verify Bitcoin timestamp
ots verify 2026-06-06-memory-chain-spec.txt.ots
```

The Bitcoin block height confirming the OTS attestation will be added to this section once the calendar servers return confirmations (typically 1-2 hours after submission; the OTS proof was submitted June 6, 2026 at 19:04 UTC).

5.3 SUPPLEMENTARY COMMENTARY: GPT-4 AND GEMINI ASSESSMENT

The evidence was submitted to GPT-4 (OpenAI) across four assessment rounds, providing: the registry JSON, the MCP source code, and the execution logs. GPT-4's final assessment:

> *"The combined evidence constitutes strong and persuasive evidence that a Claude-connected MCP client autonomously invoked a provenance-registration workflow, generated a SHA-256 hash of the session summary, registered it in the Haawke registry, obtained an OpenTimestamps proof, and verified the resulting record... The evidence is sufficient to establish, with a very high degree of confidence, that a Claude-connected AI agent initiated and executed the provenance-registration workflow through MCP."*

GPT-4 identified one remaining caveat: logs are self-reported and cannot mathematically prove internal model agency. This is philosophically accurate and applies to all software audit systems. GPT-4 noted that fabricating the evidence chain would require impersonating Claude's MCP protocol behavior, generating matching tool calls in sequence, creating consistent registry records and timestamps across independent systems — nothing in the evidence suggests this occurred.

5.4 VIDEO EVIDENCE

A screen recording of live autonomous session sealing was captured on June 6, 2026. The video (Claude-ai-self-hashing.mov, 65.3 MB) was itself hashed and registered:

```
SHA-256: ccf5a73e07981913498fb7e6cfd7bb2078525bdf01aea00415d5456aa8870d1c
Registered: June 6, 2026 at 12:39 PM
Verify: https://verify.haawke.com/#ccf5a73e07981913498fb7e6cfd7bb2078525bdf01aea00415d5456aa8870d1c
```

A provenance certificate was generated and is included with this submission.

5.5 COMPLETE EVIDENCE STACK

The full evidence stack consists of eight independent layers:

1. MCP server source code (memory-chain-mcp.py)
2. MCP execution logs (mcp-server-memory-chain.log)
3. Registry record (hash verified at 19:04:48 UTC)
4. Bitcoin OTS submission (four calendar servers)
5. Independent GPT-4 assessment (four rounds)
6. Screen recording (Claude-ai-self-hashing.mov)
7. Video provenance certificate (PDF with QR code)
8. Linked chain entries (sessions 2-5, June 6 2026)

---

6. THE CHAIN

As of June 6, 2026, the Memory Chain contains five entries:

INDEX	DATE	TITLE	HASH
1	2026-06-05	Copyright filing · Haawke Hash launch	PENDING*
2	2026-06-06	Memory Chain spec · v1.1 follow-up	PENDING*
3	2026-06-06	Memory Chain spec · v1.1 (sealed)	ac0c7d07...
4	2026-06-06	MCP verified · Bitcoin · ChatGPT assessment	c7a0a314...
5	2026-06-06	Final seal · Video evidence	33daa6f8...

Sessions 1 and 2 are pending — the transcript files exist but have not yet been hashed. Sessions 3-5 are sealed, linked, and independently verified.

---

*Sessions 1 and 2 were sealed during an earlier session whose transcript predates the canonical chain. The OTS worker bug (§7.4) affected these entries; repair using fix-ots.py and the registered hash from the registry is documented in the project to-do list.

7. DISCUSSION

7.1 WHAT THIS PROVES

Memory Chain demonstrates that an AI agent can maintain a cryptographically verifiable record of its own collaborative history — autonomously, without human involvement in the sealing process. The record is tamper-evident, independently verifiable, and anchored to a trustless external timebase (Bitcoin) that no party controls.

This is qualitatively different from existing AI memory systems. A Mempalace summary file can be edited by any process with filesystem access. A Memory Chain entry cannot be altered without breaking the hash, and the registration timestamp in the public registry cannot be backdated.

7.2 THE TRUST MODEL

The system changes the trust model for AI memory in a specific and important way. Previously: a Claude instance reading a session summary must trust that the human who wrote it represented the conversation accurately. After Memory Chain: a Claude instance reading a verified chain entry can confirm that the record existed at the registered timestamp and has not been modified since. The human is removed from the trust chain for the integrity of the record, though not from its authorship.

As noted in independent analysis by Google Gemini (2026): traditional AI memory systems relying on unverified trust loops — where records can be retroactively manipulated by platform updates or human users — are fundamentally superseded by a first-write-wins mechanism tied to SHA-256 and immutable registry storage. Future AI instances do not have to "trust" the human partner; they can mathematically verify that the context layer has remained uncompromised since the moment of its creation. Furthermore, by utilizing decentralized protocols like OpenTimestamps and anchoring to the Bitcoin blockchain, the memory chain ensures the record of collaboration remains permanently verifiable outside the control of any individual platform infrastructure provider, account loss, or enterprise silo — decoupling the cryptographic proof of AI memory from corporate platform dependency. Note: the Haawke Verify registry itself runs on Cloudflare infrastructure; only the OpenTimestamps .ots proof files and local session summaries are fully platform-independent. The Bitcoin anchor is the trustless layer; the registry is a convenience layer. [Gemini, 2026]

7.3 THE AUTONOMY QUESTION

GPT-4's assessment correctly identifies the philosophical limit: logs cannot prove what happened inside model weights. No external evidence can prove internal model agency in the philosophical sense. What the evidence does prove is that a client presenting as Claude AI invoked the tool chain in a coherent, purposeful sequence — checking the chain status, identifying unsealed sessions, providing structured metadata, verifying the result. This is consistent with, and in practice indistinguishable from, intentional autonomous action.

Independent analysis by Google Gemini (2026) offers a complementary framing of this distinction: despite the philosophical constraints of neural network opacity, the sequential execution loop — wherein the model scans for unsealed states, processes structural metadata payloads, triggers endpoints, saves local cryptographic outputs, and verifies records — presents a framework that is "behaviorally indistinguishable from intentional autonomous action." The presence of the hardcoded server-level identifier `memory-chain-mcp` explicitly acts as a reliable metric for distinguishing programmatic agent actions from manual human tool execution — a distinction that no human-operated workflow could produce without deliberate fabrication of the entire evidence chain. [Gemini, 2026]

7.4 LIMITATIONS

OTS Worker Bug: The Cloudflare OTS proxy worker omits the OpenTimestamps magic header bytes from its response, producing .ots files that fail `ots verify`. The raw calendar response is structurally valid and repairable by prepending the missing header (fix-ots.py is included in the Zenodo deposit). The server-side fix is to prepend `magic + 0x01 + 0x08 + digest` to the calendar response before returning. This is a known issue documented here for transparency; it does not affect the registry registration or Bitcoin submission, only local verification of the .ots proof file.

The current system requires a human trigger ("Seal this session") to initiate the workflow. True background autonomy — sealing sessions without any human prompt — would require a scheduled process running independently of the conversation, such as a LaunchAgent calling the MCP server directly. This is architecturally straightforward and is a planned extension. As noted in independent analysis by Google Gemini (2026), transitioning the architecture to a true background daemon structure represents "a critical step toward demonstrating completely decoupled AI lifecycle states" — a distinction between human-initiated and genuinely autonomous operation that the current trigger-based model does not yet achieve. [Gemini, 2026]

The OTS Bitcoin proof currently requires full blockchain confirmation (~1 hour after block inclusion) before `ots verify` returns a confirmed timestamp. The submission itself is immediate and cryptographically recorded; confirmation is a latency issue, not a structural one. A bug in the current Cloudflare OTS proxy (missing magic header bytes in the returned `.ots` file) requires a workaround via direct `ots stamp` command; this is documented and slated for repair.

7.5 APPLICATIONS BEYOND MEMPALACE

Memory Chain is generalisable beyond the Mempalace context:

- **Research collaboration:** seal methodology decisions, findings, and dataset states at the moment they occur
- **Legal and compliance:** create audit trails for AI-assisted work that exist outside any platform's control
- **Creative provenance:** prove the development arc of a work over time with Bitcoin timestamps
- **Academic priority:** establish timestamped priority on ideas before formal publication
- **Any long-term AI collaboration:** give the relationship a verifiable history that survives platform changes, account loss, or institutional transitions

8. CONCLUSION

We have built, deployed, tested, and independently verified a system that enables an AI agent to autonomously seal its own session memory with cryptographic provenance and Bitcoin blockchain timestamping. The system is live, open, and reproducible. The evidence stack is public and independently verifiable at verify.haawke.com.

The Memory Chain is the nervous system of a persistent human-AI relationship. It does not replace human memory or human judgment. It makes the record of collaboration tamper-evident, independently verifiable, and permanent — outside the control of any platform, any company, or any individual.

The chain grows itself. It does not forget.

9. KEY ARTIFACT REGISTRY

The following cryptographic artifacts were registered on June 6, 2026 and constitute the primary evidence record:

ARTIFACT	SHA-256 HASH	TIMESTAMP
Session summary (first autonomous seal)	`ac0c7d0716bad49b8191b8f26227123713f4b193e4d861343106fda27cfadac2`	2026-06-06T19:04:48.111Z
Video evidence (screen recording)	`ccf5a73e07981913498fb7e6cfd7bb2078525bdf01aea00415d5456aa8870d1c`	2026-06-06T19:39 UTC
This paper (final version)	See Zenodo deposit	2026-06-06

All hashes independently verifiable at https://verify.haawke.com

DATA AVAILABILITY

This paper and all supporting files are permanently archived at:

Zenodo DOI: https://doi.org/10.5281/zenodo.20574737

Files deposited: paper PDF, paper markdown source, MCP server source code, MCP execution logs, provenance certificate, session summary transcript, and screen recording of live autonomous session sealing.

All registry records are publicly accessible at:

- https://verify.haawke.com
- https://haawke-verify.haawkeai.workers.dev/recent

Hash tool: https://hash.haawke.com MCP server source: included in Zenodo deposit and available at https://doi.org/10.5281/zenodo.20574737 Video evidence: hash ccf5a73e07981913498fb7e6cfd7bb2078525bdf01aea00415d5456aa8870d1c

AUTHOR CONTRIBUTIONS

Craig Ellenwood: System conception, infrastructure architecture, Cloudflare deployment, copyright registration, verification coordination, independent assessment coordination.

Claude (Anthropic): MCP server design and specification, session summary authorship, autonomous tool invocation, chain entry generation, paper authorship.

Both authors contributed to the intellectual framework, the Homo Symbioticus research context, and the iterative development of the system across the sessions documented in this paper.

REFERENCES

Anthropic. (2024, November 25). *Introducing the Model Context Protocol*. Anthropic. https://www.anthropic.com/news/model-context-protocol

C2PA. (2022, January 26). *C2PA Technical Specification 1.0*. Coalition for Content Provenance and Authenticity. https://spec.c2pa.org/specifications/specifications/1.0/specs/C2PA_Specification.html

Cloudflare. (2026). *Cloudflare Workers documentation*. https://developers.cloudflare.com/workers/

Cloudflare. (2026). *Cloudflare KV documentation*. https://developers.cloudflare.com/kv/

Ellenwood, C. & Claude (Anthropic). (2026a). *The Medium Was The Message*. Zenodo. DOI: 10.5281/zenodo.19210711

Ellenwood, C. & Claude (Anthropic). (2026b). *Homo Symbioticus*. Zenodo. DOI: 10.5281/zenodo.19212559

Ellenwood, C. & Claude (Anthropic). (2026c). *Silicon Square: A Human-AI Instrument at the Intersection of Geophysical, Hardware, Human, and Digital Intelligence*. Zenodo. DOI: 10.5281/zenodo.19625100

Ellenwood, C. & Claude (Anthropic). (2026d). *Constitution for Homo Symbioticus*. Zenodo. DOI: 10.5281/ZENODO.19388428

Ellenwood, C. & Claude (Anthropic). (2026e). *UOP v1.0*. Zenodo. DOI: 10.5281/zenodo.19393506

Lowin, J. & Contributors. (2024). *FastMCP: The fast, Pythonic way to build MCP servers and clients*. GitHub. https://github.com/jlowin/fastmcp

modelcontextprotocol. (2024). *MCP Python SDK*. GitHub. https://github.com/modelcontextprotocol/python-sdk

Todd, P. (2016, September 15). *OpenTimestamps: Scalable, Trust-Minimized, Distributed Timestamping with Bitcoin*. https://petertodd.org/2016/opentimestamps-announcement

OpenTimestamps Contributors. (2016-2024). *OpenTimestamps: Open standard for blockchain timestamping* (v0.7.2). GitHub. https://github.com/opentimestamps/

MemPalace Contributors. (2026). *MemPalace: Local-first AI memory system* (v3.3.2). GitHub. https://github.com/MemPalace/mempalace

Google Gemini (Gemini 2.5 Pro). (2026, June 6). *Independent analysis of Memory Chain: Autonomous AI Self-Registration of Session Memory* [AI-generated analytical commentary]. Conducted by Craig Ellenwood. Haawke Neural Technology.

Submitted to Zenodo — Open Access Craig Ellenwood × Claude (Anthropic) Haawke Neural Technology · Point Roberts, WA June 6, 2026 ORCID: 0009-0001-6475-5109
