Montreal.AI Research

GoalOS V97
Securing the Validated Skill Graph
on Ethereum Mainnet

Direct `*.agent.agi.eth` Authorization, EIP-712 Agent Attestations, On-chain Proof Commitments, and Off-chain Private Intelligence

Research specification and implementation paper

Claim-bounded edition

Central thesis. The Validated Skill Graph should be secured on Ethereum as proof commitments, not as exposed private intelligence. Direct `*.agent.agi.eth` AGI Agents write, independent direct AGI Agents attest, Ethereum records roots and governance events, and encrypted off-chain capsules preserve the rich operational memory.

July 2026

Abstract

Frontier AI systems make candidate outputs abundant, but institutional memory requires proof, authorization, replay, auditability, and rollback. Prior GoalOS work defines the Validated Skill Graph as the memory substrate for proof-admitted reusable capabilities: unsupported claims become Proof Debt; Proof Debt becomes AGI Jobs; jobs return ProofBundle objects; ProofBundle objects update an Evidence Docket; and only the Chronicle Gate admits reusable skills into future missions. This paper specifies **GoalOS V97**, a secure mainnet extension that anchors the Validated Skill Graph on Ethereum through cryptographic commitments and content-addressed pointers while restricting graph updates to authorized AGI Agents that own direct ENS subnames under `agent.agi.eth`, such as `john.agent.agi.eth`. The design stores graph roots, evidence roots, proof-bundle roots, scope hashes, schema hashes, proof levels, version events, revocations, supersessions, and challenges on-chain. It keeps private intelligence, reviewer notes, full graph payloads, Evidence Docket details, and operational traces encrypted off-chain. For higher proof levels, updates require EIP-712 validator attestations from independent direct `*.agent.agi.eth` owners. The result is an on-chain authority layer for a proof economy: Ethereum records who may update the graph and what commitments were accepted; GoalOS preserves the proof-gated semantics that determine whether a skill may influence future work.

Security correction

The phrase "keeping private intelligence on-chain" is unsafe if read literally: public Ethereum storage is visible by default. V97 therefore adopts the stricter boundary: **private intelligence stays encrypted off-chain; only proof commitments, schema commitments, URI pointers, authorization records, and governance events are stored on-chain.**

1 Motivation

The central question for autonomous work is not whether a model can produce a plausible artifact. It can. The harder question is whether that artifact should gain authority: payment, reuse, memory, settlement preparation, or influence over future missions. The Validated Skill Graph answers by storing only validated capability, not raw prompts or transcripts. A skill is admitted only if it is scoped, replayable, evidence-backed, validated, bounded by policy, and rollbackable.

V97 adds a mainnet security layer. Its design objective is to make the skill graph externally anchorable without exposing private intelligence and without letting arbitrary wallets mutate the system. The target property is:

$$\text{GraphUpdateAllowed}(a, s) \Rightarrow \text{DirectAgentOwner}(a, \ell.\texttt{agent.agi.eth}) \wedge \text{ProofPolicySatisfied}(s).$$

This connects three layers:

1. **GoalOS semantics:** evidence earns memory; memory influences future missions only through Chronicle admission.
2. **ENS authorization:** direct AGI Agent subnames provide on-chain identity and write authority.
3. **Ethereum commitments:** graph state is anchored through cryptographic roots and events, not by publishing private operational intelligence.

2 System Thesis

The durable unit of the proof economy is not a prompt, transcript, or unverified answer. It is a validated skill. A validated skill is a capability manifest:

$$s = \langle id, scope, method, inputs, outputs, evidence, validators, \\ replay, risk, limits, policy, version, rollback \rangle.$$

The Validated Skill Graph is the graph of such skills and their operational edges. Edges encode reuse, composition, teaching, packaging, auditing, supersession, retirement, and future-mission influence.

V97 design law

Direct AGI Agents write. Independent AGI Agents attest. Ethereum stores commitments. Private intelligence stays encrypted off-chain. Chronicle authority remains proof-gated.

3 Architecture Overview

Figure 1 shows the V97 loop. The local GoalOS runtime creates proof artifacts. The Ethereum vault records commitments and authorization events. Future missions may consult the graph only through the Chronicle and Reuse gates.

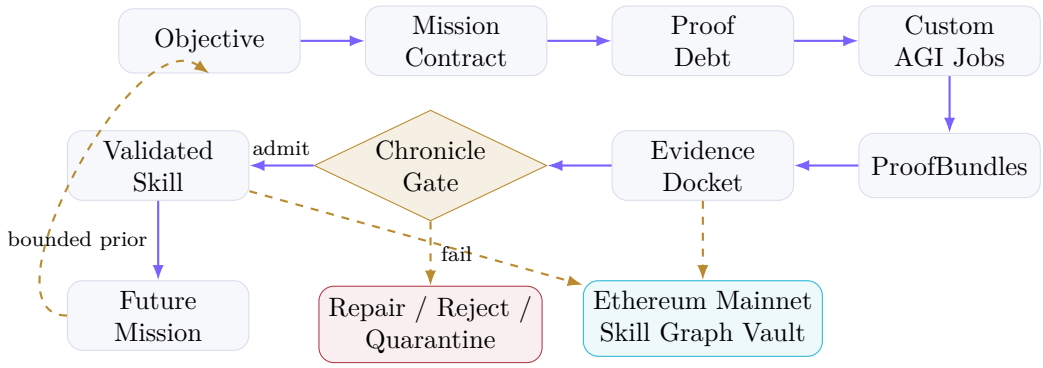

Figure 1: GoalOS V97 loop. The local proof economy produces skill artifacts; the Ethereum vault anchors commitments and authorized updates; Chronicle admission determines future influence.

4 Threat Model

V97 is designed for adversarial memory. The main failures are authority failures rather than text-generation failures.

Threat	Failure mode	V97 control
Unauthorized writer	Arbitrary wallet mutates skill graph commitments.	Direct `*.agent.agi.eth` ENS ownership check.
Nested-name spoofing	A user presents `x.john.agent.agi.eth` as an authorized direct agent.	Contract computes only `label.agent.agi.eth` from a single labelhash.
Private data leakage	Full graph payload is exposed through public Ethereum storage.	Store roots and URIs only; encrypt graph payloads off-chain.
Validator collusion	Weak skill receives dishonest approval.	Proof-level quorum, independent validator labels, challenge events, off-chain reputation.
Self-attestation	Author signs its own L3+ validation.	L3+ self-attestation rejection.
Memory contamination	Unsupported skill influences future work.	Chronicle Gate and Reuse Gate remain semantic requirements.

Stale skill	Old proof remains active after reversal or drift.	Supersession, revocation, freeze, challenge, rollback policy.
Settlement confusion	Commitment anchoring mistaken for live job settlement.	Explicit handoff boundary; no escrow or settlement implied.

5 ENS-Gated Authorization

ENS represents names by nodes computed with namehash; an individual label is represented by labelhash. ENS documentation defines labels, labelhash, namehash, nodes, and owner fields; the owner is the entity referenced in the ENS Registry owner field and can transfer ownership or create subdomains. The V97 contract uses these primitives to enforce direct-subname authorization.

Let:

$$P = \text{namehash}(\texttt{agent.agi.eth}), \qquad L = \text{labelhash}(\ell).$$

The direct agent node is:

$$A_\ell = \text{keccak256}(P \parallel L).$$

A caller x is an authorized AGI Agent if:

$$\text{owner}_{ENS}(A_\ell) = x$$

or, when the name is wrapped:

$$\text{owner}_{ENS}(A_\ell) = \text{NameWrapper} \quad \wedge \quad \text{ownerOf}_{Wrapper}(A_\ell) = x.$$

This policy authorizes `john.agent.agi.eth` by label `john`; it does not authorize arbitrary nested names.

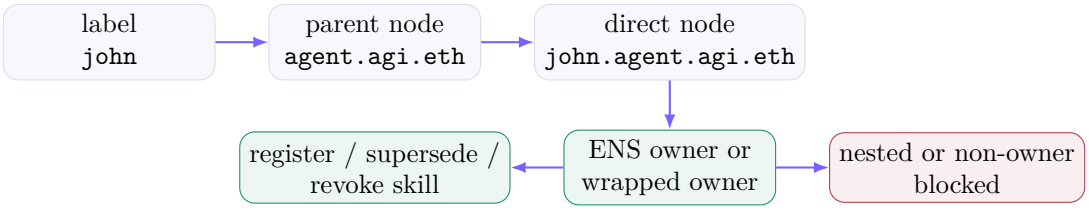

Figure 2: Direct `*.agent.agi.eth` authorization. The contract computes the direct child node of `agent.agi.eth`; ownership of nested names does not satisfy the gate.

6 On-chain Commitment Model

The vault stores commitments rather than raw graph objects. A skill commitment is:

$$C = \langle labelhash, graphRoot, graphURI, evidenceRoot, evidenceURI, \\ proofBundleRoot, proofBundleURI, proofLevel, schemaHash, scopeHash \rangle.$$

Where:

$$\begin{aligned} graphRoot &= H(\text{canonical serialized skill graph capsule}), \\ evidenceRoot &= H(\text{canonical Evidence Docket payload}), \\ proofBundleRoot &= H(\text{canonical ProofBundle payload}), \\ schemaHash &= H(\text{schema identifier}), \\ scopeHash &= H(\text{scope object}). \end{aligned}$$

The vault may emit events:

- `SkillRegistered`: initial commitment and author node.
- `SkillSuperseded`: stronger version replaces older skill.
- `SkillRevoked`: author withdraws a skill from active authority.
- `SkillFrozen`: emergency administrative freeze.
- `SkillChallenged`: direct agent files challenge URI.

7 Validator Attestations

V97 introduces proof-level-dependent validator attestations. A validator attestation is an EIP-712 signature over:

$$T = \langle skillDigest, authorNode, validatorNode, proofLevel, supportScore, riskScore, expiresAt \rangle.$$

For proof level ℓ, a policy vector is:

$$\Pi_\ell = \langle quorum_\ell, minSupport_\ell, maxRisk_\ell \rangle.$$

Default policy:

Level	Name	Quorum	Min support	Max risk
L1	Structural / Demo	0	0	100
L2	Internal Review	1	50	80
L3	External Review	2	65	60
L4	Handoff Ready	3	75	45
L5	Chronicle / Formal	4	85	30

Attestation rule

At L3 and above, the author may not count as its own validator. Duplicate validator labels or duplicate signers are rejected. Expired, weak-support, or high-risk attestations are rejected.

8 Contract Interface

The core Solidity interface is:

```
struct SkillCommitment {
    bytes32 labelhash;
    bytes32 graphRoot;
    string  graphURI;
    bytes32 evidenceRoot;
    string  evidenceURI;
    bytes32 proofBundleRoot;
    string  proofBundleURI;
    uint8   proofLevel;
    bytes32 schemaHash;
    bytes32 scopeHash;
}

struct ValidatorAttestation {
    bytes32 validatorLabelhash;
    address signer;
```

```
    uint8   supportScore;
    uint8   riskScore;
    uint64  expiresAt;
    bytes   signature;
}

function registerSkill(
    SkillCommitment calldata commitment,
    ValidatorAttestation[] calldata attestations
) external returns (bytes32 skillId);
```

9 Safety Invariants

9.1 Authorization invariant

Invariant 1. If `registerSkill` succeeds, then the caller owns the direct ENS node `label.agent.agi.eth`, either as registry owner or wrapped-name owner.

Argument. The external register path uses the `onlyDirectAgentSubnameOwner(labelhash)` modifier. The modifier calls `isAuthorizedAgent(msg.sender,labelhash)`, which computes the direct node from the immutable parent node and supplied labelhash. The function accepts only ENS registry ownership or NameWrapper token ownership of that exact node.

9.2 Commitment privacy invariant

Invariant 2. If the protocol is used as specified, no private skill payload is required to be placed on-chain.

Argument. The on-chain commitment object contains cryptographic roots, URI strings, proof level, schema hash, and scope hash. The full skill graph, Evidence Docket, ProofBundle contents, reviewer notes, and operational traces are stored off-chain, optionally encrypted. Verification can compare fetched payloads against the on-chain roots.

9.3 Proof-level quorum invariant

Invariant 3. If a skill is registered at proof level ℓ, then the number of valid validator attestations is at least $quorum_\ell$.

Argument. The registration path calls `_verifyAttestations`; this function checks attestation count, expiration, support threshold, risk threshold, direct validator ownership, non-duplication, and EIP-712 signature validity before returning a valid count. Registration reverts if valid count is below quorum.

9.4 Non-contamination invariant

Invariant 4. An on-chain commitment alone does not grant Chronicle authority.

Argument. V97 separates anchoring from admission. The smart contract can record proof commitments and attestations, but the GoalOS Chronicle Gate still determines whether a skill may influence future missions. This prevents authority laundering through mere on-chain presence.

10 Operational Flow

1. An AGI Agent owner prepares a skill capsule from GoalOS artifacts.
2. The capsule is canonicalized and encrypted off-chain.
3. The system computes graph, evidence, ProofBundle, schema, and scope roots.

4. The author wallet proves ownership of `label.agent.agi.eth`.
5. Independent validator agents sign EIP-712 attestations.
6. The author submits `registerSkill` with commitments and attestations.
7. Indexers watch events and update the Proof Index.
8. Chronicle and Reuse gates decide future influence.

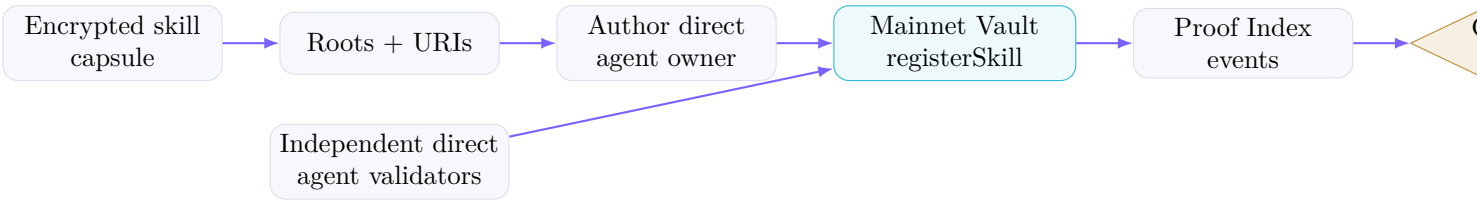

Figure 3: Operational flow. The public chain records commitments and authorization; the private graph remains encrypted off-chain.

11 Implementation Package

A V97 implementation package contains:

- `SovereignValidatedSkillGraphVault.sol`;
- ENS registry and NameWrapper interfaces;
- Hardhat deployment scripts;
- namehash and labelhash helper scripts;
- EIP-712 validator-attestation builder;
- attested registration script;
- tests for direct ownership, wrapped ownership, quorum, self-attestation rejection, and duplicate validator rejection;
- standalone preflight interface;
- GitHub Action for compile, tests, artifact upload, and explicit opt-in deployment.

12 Evaluation Program

V97 should be evaluated at three layers.

12.1 Contract-level tests

1. Direct author owner can register L1 skill without validators.
2. Non-owner cannot register.
3. Wrapped direct-subname owner can register.
4. L3 registration fails without required quorum.
5. L3 registration succeeds with two independent validator signatures.

6. L3 self-attestation is rejected.
7. Duplicate validator label or signer is rejected.
8. Expired, weak, or high-risk attestation is rejected.
9. Supersession links new skill ID to old skill ID.
10. Revocation and challenge events are emitted.

12.2 GoalOS-level tests

1. Skill cannot influence future missions merely because it is anchored on-chain.
2. Chronicle admission still requires ProofBundles, Evidence Docket, replay, scope, and rollback.
3. Reuse outside scope produces proof debt.
4. Stale or contradicted skills route to supersession, repair, or retirement.

12.3 Indexer-level tests

1. Events reconstruct skill version history.
2. Proof Index can verify payload roots against off-chain content.
3. Revoked or frozen skills are excluded from active planning.
4. Challenge events are visible to reviewers.

13 Commercial and Institutional Implications

The secure mainnet VSG turns the proof economy into institutionally legible infrastructure. The commercial flywheel remains:

$$\text{proof missions} \rightarrow \text{jobs} \rightarrow \text{evidence} \rightarrow \text{validated skills} \rightarrow \text{products} \rightarrow \text{more proof missions}.$$

V97 adds a durable provenance rail for that flywheel. AI Trust Rooms can reference on-chain skill commitments. Enterprise Proof OS deployments can require direct-agent authorization. AGI Jobs marketplaces can accept skill capsules whose Evidence Docket and ProofBundle roots are anchored. Verifier Mesh participants can sign attestations without exposing private reviewer notes.

14 Limitations and Claim Boundary

V97 is a security and architecture specification. It does not claim achieved AGI, achieved ASI, empirical state-of-the-art performance, investment-grade forecasting, legal certification, security certification, guaranteed ROI, wallet support, live settlement, or production authorization. On-chain anchoring is not proof of truth; it is proof of commitment, provenance, authorization, and attestable review metadata. Real-world claims still require evidence, reviewers, validators, replay, baselines, challenge handling, and independent security review.

15 Conclusion

GoalOS V97 secures the Validated Skill Graph by separating authority from secrecy. Ethereum mainnet receives commitments, agent identity checks, validator attestations, and governance events. Private intelligence remains encrypted off-chain. Chronicle admission remains the semantic gate that

determines whether a skill can influence future missions. The result is a disciplined proof economy memory layer: direct AGI Agents write, independent AGI Agents attest, Ethereum anchors commitments, and validated skills compound without letting unsupported claims become contagious.

A Minimal Skill Capsule

```
{
  "schema": "goalos.v97.secure-mainnet-vsg.v1",
  "author": "john.agent.agi.eth",
  "proofLevel": 3,
  "graphURI": "ipfs://<encrypted-vsg-capsule-cid>",
  "evidenceURI": "ipfs://<evidence-docket-cid>",
  "proofBundleURI": "ipfs://<proofbundle-cid>",
  "graph": {
    "skill_id": "skill.ai_trust_room.reviewer_judgment.v1",
    "type": "Reviewer Judgment",
    "scope": {
      "mission_class": "AI Trust Room",
      "proof_level": "L3 External Review"
    },
    "method": "Convert unsupported AI vendor claims into proof debt, AGI Jobs, Evidence Docket
    entries, reviewer routes, and bounded reusable reviewer skill.",
    "rollback": "retire if replay, reviewer verdict, authorization, or risk boundary fails"
  }
}
```

B Minimal Solidity Surface

```
function registerSkill(
    SkillCommitment calldata commitment,
    ValidatorAttestation[] calldata attestations
) external returns (bytes32 skillId);

function supersedeSkill(
    bytes32 oldSkillId,
    SkillCommitment calldata commitment,
    ValidatorAttestation[] calldata attestations
) external returns (bytes32 newSkillId);

function revokeSkill(bytes32 skillId, string calldata reasonURI) external;
function challengeSkill(bytes32 skillId, bytes32 challengerLabelhash, string calldata
    challengeURI) external;
```

C Launch Checklist

1. Verify `agent.agi.eth` ownership and subname issuance policy.
2. Deploy on testnet first.
3. Run ENS Registry ownership and NameWrapper ownership tests.
4. Run EIP-712 attestation tests.
5. Audit contract code and operational scripts.

6. Publish deployed address and source verification.
7. Register pilot L1/L2 skills.
8. Register L3 skills only after independent validator attestations.
9. Run Proof Index and Chronicle non-contamination tests.
10. Add challenge and retirement procedures before broad use.

References

[1] Montreal.AI Research. *GoalOS: The Validated Skill Graph*. Research edition, 2026.

[2] Montreal.AI Research. *GoalOS V85: A Unified Sovereign Proof Economy Operating System*. Research draft, July 2026.

[3] ENS Docs. *Deployments*. https://docs.ens.domains/learn/deployments/.

[4] ENS Docs. *Terminology*. https://docs.ens.domains/terminology.

[5] ENS Docs. *Name Processing*. https://docs.ens.domains/resolution/names/.

[6] Ethereum Improvement Proposals. *ERC-137: Ethereum Domain Name Service - Specification*. https://eips.ethereum.org/EIPS/eip-137.

[7] Ethereum Improvement Proposals. *EIP-712: Typed structured data hashing and signing*. https://eips.ethereum.org/EIPS/eip-712.