ULI: Sovereign Identity with Zero-Knowledge Credentials

Timothy Bobinsky
timothy@uli.foundation
June 2026

Abstract. ULI (Unified Ledger Identity) is a Layer-1 blockchain providing recoverable, key-independent identity with zero-knowledge credentials and a privacy-layered social graph. Each participant (human, organisation, or autonomous agent) receives a persistent identity with mutable keys, three recovery mechanisms, and scoped delegation for AI agents. Groth16/BN254 credentials let applications verify properties (age, personhood, institutional membership) without revealing underlying attributes; a global nullifier set prevents double-use. Users never hold tokens: application operators stake capacity and sponsor transactions via fiat-pegged Capacity Credits. Consensus is Fast-HotStuff BFT with BLS12-381 aggregation; content and social edges reside on an erasure-coded DA layer.

1. Introduction

Digital identity has become the critical infrastructure layer for applications across finance, social media, and autonomous AI systems. Yet existing approaches force a choice between mutually desirable properties. Biometric systems achieve strong uniqueness but produce irrevocable commitments whose trust assumptions cannot be updated after deployment. Platform-controlled accounts offer convenience but lock users into opaque data silos. Key-based protocols ($\mathcal{I} \equiv \mathcal{K}$) maximise decentralisation but sacrifice recoverability: a lost key means a lost identity. Partial solutions exist (NEAR Protocol separates identity from keys, ERC-4337 wallets add contract-level recovery), but none integrates key-independent identity, protocol-level recovery, zero-knowledge credential verification, and privacy-preserving social graph attestation in a single system.

ULI is identity infrastructure that resolves this trade-off. The economic model has two sides: social applications onboard users cheaply (batched graph anchoring, negligible per-user cost), and identity verification (credential checks, nullifier operations, agent attestation) generates per-verification fee burn that scales with verification volume. Social adoption builds the graph; verification sustains the security budget. Each participant (human, organisation, or autonomous agent) receives a persistent identifier (ULI ID) bound to a mutable set of cryptographic keys, recoverable through three complementary mechanisms (social guardians, timelock backup, FROST threshold signing), and extensible through zero-knowledge credentials that reveal nothing beyond the specific claim being proven. The protocol serves as a substrate for applications:

1. **Verifiable credentials without PII exposure.** An application verifies "attribute $\geq$ threshold" or set membership via a Groth16 proof; the user reveals nothing else. The same credential is reusable across services via domain-scoped nullifiers.
2. **Agent and bot attestation.** AI agents operate under cryptographic delegation chains with per-operation spending limits, providing auditable authorization without shared secrets. Applications distinguish *human-attested*, *bot-declared*, and *delegated-agent* accounts at the protocol level.
3. **Privacy-preserving social graph.** Social relationships are encrypted per-edge and provable via zero-knowledge proofs: a user can demonstrate "I have N vouches from the honest core" without revealing which accounts vouched for them.

Application operators (providers) stake tokens and sponsor user transactions via a capacity model (Section 8), so end users never interact with cryptocurrency.

Three separations. The technical contribution of this work is the realisation of three separations that existing protocols conflate:

1. **Identity from keys.** $\mathcal{I} \not\equiv \mathcal{K}$: an identity maps to a mutable set of keys, not the reverse. Key compromise triggers recovery, not identity loss.
2. **Content from consensus.** Consensus processes identity operations and hash-anchors; bulk data resides on a data-availability layer.
3. **Privacy from transparency.** Zero-knowledge proofs protect specific attributes without requiring the entire chain to operate inside a ZK circuit.

Layered uniqueness model. The protocol deliberately *does not* enforce uniqueness at the consensus layer. One person may hold multiple ULI IDs; one ULI ID may represent a bot, a group, or a pseudonym. Uniqueness is an *application-level* property, and it operates in two tiers:

1. **Credential-based uniqueness** (cryptographic, strong). An external issuer (government, KYC provider, biometric service) certifies a credential; the domain-scoped nullifier guarantees that this credential cannot be presented twice within the same application scope. The chain provides the global, censorship-resistant nullifier set; the strength of the uniqueness guarantee comes from the issuer, not from the chain. This tier serves regulated use cases (banking, subsidies, governance) and provides the hard guarantee for AI agent accountability: proving that a delegated agent is bound to a verified human requires an externally issued personhood credential.
2. **Graph-based Sybil resistance** (indexer layer, probabilistic). Social vouching and graph-based Sybil detection (Section 6) provide a different assurance model: continuous, privacy-preserving at the verifier level (ZK proof of score, not graph access), and strengthening with genuine social interaction, but probabilistic, not absolute. High trust scores indicate likely uniqueness; small-scale collusion (2–3 accounts) can evade detection. Trust scoring is computed by staked indexers whose results are fraud-provable on-chain (Section 6), not by unverifiable off-chain computation. This tier is sufficient for spam prevention, community gating, and social reputation, but not for use cases that require a hard uniqueness guarantee.

Applications choose their tier based on the strength they need. A social platform may accept graph-based personhood; a higher-assurance application may require an externally issued credential; an agent registry may require a credential proving that a controlling human exists. In both cases, the nullifier prevents double-use of whichever credential the application accepts. The chain is *credential verification infrastructure*: it makes strong issuers *portable* and *privacy-preserving* via ZK proofs and a global nullifier set, and provides verifiable trust scoring via the social graph (backed by staked indexer commitments) for contexts where a probabilistic signal suffices.

Design space. Section 10 analyses existing approaches in detail. Three structural gaps motivate ULI's design:

- *Recovery.* No deployed decentralised identity protocol provides protocol-level key recovery. Nostr, Farcaster, and AT Protocol all equate identity with a cryptographic key; a lost key means starting over.
- *Personhood without irrevocable biometrics.* Worldcoin achieves strong uniqueness via iris biometry but creates an irrevocable commitment that has been banned or suspended in multiple jurisdictions on data-protection grounds [12]. Biometric collection at scale faces growing legal and social resistance; many markets need an alternative path to personhood scoring that trades biometric uniqueness for broad deployability.
- *Private graph with ZK attestation.* AT Protocol's social graph is public by design; block lists are visible to all. Applications that require anti-Sybil or anti-fraud analysis over relationship data (without exposing that data) have no existing solution.

ULI addresses all three: three recovery paths at the protocol level, personhood via social vouching with optional external attestation, and a privacy-layered social graph provable via zero-knowledge proofs.

Agent attestation. The growth of AI agents, autonomous systems that transact, post, and interact on behalf of humans, creates demand for cryptographic categorisation that existing identity systems do not provide. As agents proliferate across commerce, social media, and governance, platforms need to answer: *is this entity a verified human, a declared bot, or a delegated agent acting under human authority?* ULI defines three account categories: *human-attested* (backed by an externally issued personhood credential or, at lower assurance, by graph-based Sybil resistance), *bot-declared* (self-identified as automated), and *delegated-agent* (operating under a derived key with per-operation spending limits from a parent human account). Applications filter by category: a forum may demand a personhood proof for human-only threads; a trading interface may accept bot-declared accounts; a high-assurance application may combine human attestation with an externally issued credential. Delegation chains are cryptographically auditable: every agent action traces to the human who authorised it, with spending limits and revocation enforced on-chain.

The remainder of this paper describes the system model (Section 2), identity layer (Section 3), consensus protocol (Section 4), transaction format (Section 5), Anchored Social Graph Protocol (Section 6), zero-knowledge credential system (Section 7), economic model (Section 8), moderation (Section 9), and related work (Section 10).

2. System Model

ULI is organised into three layers with distinct responsibilities (Figure 1).

Consensus Layer (L1). A BFT state machine replicated across validators. It processes identity operations (account creation, key rotation, recovery, delegation), validator staking, schema registration, zero-knowledge credential verification, and hash-anchors for off-chain data. Content is *never* stored in consensus state; only its Blake3 hash enters the chain.

Data-Availability Layer. Stores social-graph edges, post bodies, encrypted direct messages, and media references. Each piece of content is referenced on-chain by a `DaReference` and cryptographically bound by a Blake3 content hash. The primary DA backend is a sovereign P2P blob store with erasure coding; capacity providers replicate blobs for the accounts they sponsor. Validators enforce DA validation during consensus voting: a validator withholds its BLS vote for any block whose DA-referencing transactions contain unretrievable blobs, so a quorum certificate constitutes a collective attestation of data availability by $> 2S/3$ stake. Long-term storage beyond the GC window ($\approx$7 days) is the provider's responsibility (see [22] for retention economics).

Indexers. Stateless read replicas that consume the L1 block stream and DA blobs, reconstruct the full social graph in a relational database, and expose an HTTP API. Multiple independent indexers may operate concurrently, each implementing its own ranking and filtering algorithms. Indexers are not trusted for writes: all mutations flow through signed transactions submitted to L1.

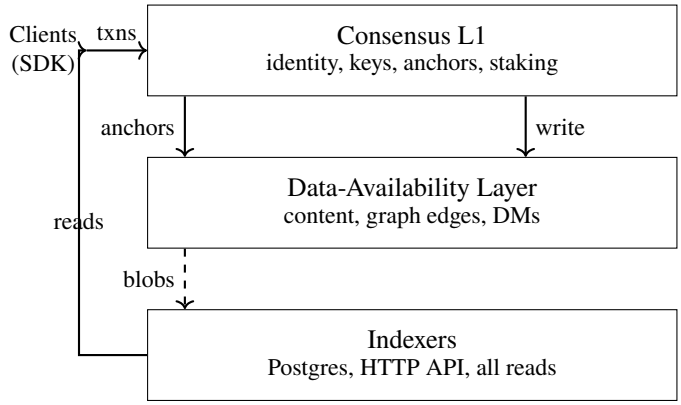

Figure 1: Three-layer architecture. Clients submit transactions to L1 and read from indexers. Content flows through the DA layer; only hash-anchors enter consensus.

The separation ensures that consensus processes only lightweight hash-anchors and identity operations, not bulk content data. The analytical block-capacity maximum is 426 TPS (256 transactions per block at 600 ms slots, ~13.5 billion anchors per year). Single-node load testing (1 s blocks, 256 tx/block cap, onboarding workload) measured 253 committed TPS at 97% average block fill with zero rejected transactions: throughput is bounded by block capacity rather than execution. Sustained throughput under BFT with geographically distributed validators will be bounded by network round-trip time and QC aggregation latency; geo-distributed benchmarks are pending testnet deployment. Applications requiring higher throughput can batch multiple content items into a single graph-root update via `AnchorGraph`.

Table 1 summarises consensus-critical parameters. Formal definitions of all system model components are provided in the technical specification [22].

Parameter	Value
Max transactions per block	256
Max ZK transactions per block	32
Max credentials per account	64
Max derived keys per account	32
Max retired keys	16
Max guardians (social recovery)	8
Max unbonding entries	32
Max labeler subscriptions	64
Slot duration	600 ms
Slots per epoch	432,000
Social recovery timelock	1,008,000 slots
Min timelock recovery	432,000 slots
Min jail duration	7 epochs
Min provider stake	1,000 tokens
Provider grace period	7 epochs
Min validator self-stake	10,000 tokens
Capacity per token	10 units
Oracle TWAP window	7 epochs
Labeler deposit	10,000 tokens
Governance quorum	33%
Governance threshold	67%
Equivocation slash	10%

Table 1: System parameters.

3. Identity

3.1 ULI ID

Each account is identified by a ULI ID: a monotonically increasing `u64` counter assigned at account creation. The identifier never changes, regardless of key rotation, recovery, or device migration. Each ULI ID resolves to a W3C-compatible DID: `did:uli:<id>`. The DID document exposes the account's active public key, recovery configuration, and service endpoints, enabling interoperability with W3C Decentralized Identifiers [20] and enterprise identity wallet ecosystems. No personally identifiable information is stored on-chain.

The identity-key separation ($\mathcal{I} \not\equiv \mathcal{K}$) resolves a class of problems that plague key-based identity systems: key compromise does not require social-graph reconstruction, key rotation does not break existing references, and multiple devices can share identity through derived keys rather than sharing a single private key.

3.2 Key Management

Account keys use Sr25519 (Schnorr signatures on Ristretto255 [10]), chosen for direct compatibility with FROST-Ristretto [6] threshold signing. The key registry maintains a single active key and up to 16 retired keys. A reverse index maps each public key to its account ID, enabling $O(1)$ lookup during transaction validation.

3.3 Derived Keys and Agent Attestation

An account holder may authorise up to 32 derived keys, each scoped by a `SpendingLimit` with permission mode (AllowList or DenyList), global token limit, per-operation quotas, and an expiration block. Derived keys enable applications to act on behalf of users without holding the owner key, analogous to OAuth scopes but enforced at the consensus layer.

The protocol defines three account categories:

1. **Human-attested**: holds a valid personhood credential (Section 7) verified via ZK proof.
2. **Bot-declared**: self-identified as automated via voluntary on-chain attestation.
3. **Delegated agent**: operates under a derived key with spending limits from a parent account; the delegation chain is auditable on-chain.

This mechanism serves as a cryptographic authorization layer for AI agents: an AI platform issues a derived key to each agent instance with scoped permissions and a token budget, providing auditable, revocable authority without shared secrets.

3.4 Recovery

Three complementary recovery mechanisms prevent instant account takeover while remaining practical for legitimate recovery (Figure 2). In practice, recovery paths may share failure modes: guardians are typically drawn from the account holder's social circle (correlated with social-engineering risk), FROST key-share holders may overlap with guardians, and backup key storage discipline correlates with primary key discipline. Users should diversify recovery paths across distinct trust domains to maximise effective redundancy.

Social Recovery. Up to 8 guardians and a threshold t; recovery requires t approvals followed by a 7-day timelock (1,008,000 slots). Cancellation behaviour during the waiting period is governed by a per-account policy (see below). Guardian-set changes (`SetSocialRecovery`) are themselves subject to a timelock, preventing an attacker from silently removing guardians before initiating recovery.

Timelock Recovery. A pre-registered backup key with a minimum 3-day waiting period (432,000 slots). The same per-account cancellation policy applies.

Threshold Recovery. FROST-based [6] t-of-n threshold signing with no timelock; security derives from the threshold itself.

Cancellation policy. The owner key and a stolen key are indistinguishable on-chain: if cancellation is allowed during the timelock, a key thief can veto every legitimate recovery; if it is blocked, a malicious guardian coalition completes an irrevocable takeover. The protocol makes this a per-account choice set in `SetSocialRecovery` (changeable only through a timelock): *KeyWins* (default) preserves the owner's veto during the waiting period, with FROST as the recommended fallback for key-theft scenarios; *QuorumWins* blocks cancellation once the timelock

begins, protecting legitimate recovery from key-thief interference. Reconfiguration (key rotation, guardian removal, recovery-config changes) is unconditionally blocked during any active recovery, regardless of policy.

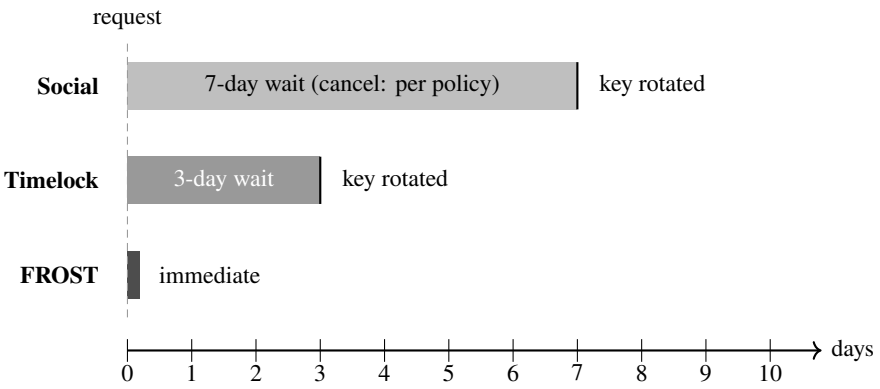

Figure 2: Recovery timelines. FROST threshold recovery (t-of-n) completes immediately; timelock and social recovery impose mandatory waiting periods with per-account cancellation policy (*KeyWins* or *QuorumWins*).

3.5 Standards Compatibility

Each `did:uli:<id>` resolves to a DID Document containing the account's active public key, recovery methods, and service endpoints. A bridge layer maps ULI ZK credentials to W3C Verifiable Presentation format, allowing enterprise systems that consume OID4VP to accept ULI credentials without modification. This positions ULI as infrastructure *beneath* enterprise identity wallets rather than a competing standard.

Demand source: credential verification and agent attestation. The core demand for ULI does not depend on any specific regulation. It arises from two structural trends.

First, when multiple independent parties need to verify that the same person has not already claimed a benefit, cast a vote, or opened an account elsewhere, they need a shared nullifier set that no single party controls. A centralised registry requires trusting the operator not to censor, forge, or selectively disclose nullifiers. A federated approach partitions the set, destroying global uniqueness. Only a neutral, append-only, censorship-resistant log, a blockchain, can serve as the canonical nullifier store.

Second, the proliferation of AI agents that transact and interact on behalf of humans creates demand for cryptographic accountability: binding every agent action to a verified human via auditable delegation chains, anchored beyond any single platform.

These trends drive four concrete use cases:

- **One-person-one-account.** Social platforms, governance systems, and airdrop distributions need proof that a user is unique across the system, not just within one service. For social platforms, graph-based Sybil resistance (probabilistic, sufficient for spam and reputation) is a cheap first filter; for regulated governance, an externally issued credential provides the hard guarantee.
- **AI agent accountability.** Proving that an autonomous agent is bound to a verified human (not a bot farm) requires a tamper-evident delegation chain anchored to a credential-backed personhood proof. Graph-based scoring cannot serve this role: a motivated adversary can simulate social interactions; only an external credential provides the assurance level agent accountability requires.
- **Portable credential verification.** A credential issued by party A (bank, university, employer) should be verifiable by independent parties B, C, D without contacting A again, and without allowing the holder to present it as two different people. The nullifier prevents double-use; the issuer's certification provides the uniqueness guarantee.
- **Subsidy and benefit verification.** Programmes that distribute per-person benefits require strong uniqueness from the issuing authority; the nullifier ensures each certified beneficiary claims once.

In every case, the nullifier ensures that the issuer's certification is consumed exactly once per domain (Section 1, "Layered uniqueness model"). The chain does not make weak issuers strong; it makes strong issuers *portable* and *privacy-preserving*. The social graph provides verifiable trust scoring, backed by staked indexer commitments with fraud proofs, for contexts (spam, community gating) where a probabilistic signal suffices and an external credential would impose unnecessary friction.

Regulatory trends toward privacy-preserving digital identity, including data-protection laws that constrain biometric data collection, are a tailwind that accelerates demand for credential infrastructure. But ULI's demand thesis does not depend on any specific mandate. The protocol serves any domain where credential verification and identity portability are needed (social platforms, fintech, community governance, AI agent registries) regardless of whether a particular jurisdiction mandates it by statute. First-party social applications and institutional pilot partnerships (Section 8) provide the initial demand channel independently of regulatory timelines.

4. Consensus

ULI uses Fast-HotStuff [2], a refinement of HotStuff [1] that reduces the commit rule from 3 phases to 2 while preserving $O(n)$ message complexity per view. Vote aggregation uses BLS12-381 [3] signature aggregation.

The protocol operates in numbered views. Each view, a deterministic leader proposes a block of up to 256 transactions; validators verify and vote; votes are aggregated into a quorum certificate (QC) requiring $> 2S/3$ stake. A block commits when the next consecutive view carries a valid QC for it (2-chain commit rule).

Safety follows from quorum intersection: any two quorums overlap in $> S/3$ stake, exceeding the Byzantine budget under $n \geq 3f + 1$. Liveness is guaranteed under partial synchrony via timeout certificates. Equivocation (signing two blocks in the same view) triggers a 10% stake slash and 7-epoch jail.

Protocol details and safety proof: see [22], Section 4.

5. Transactions

Table 2 summarises the 49 transaction types across twelve categories. Transactions are signed with Sr25519, include a monotonic nonce and chain identifier (preventing replay), and are hashed with Blake3.

Category	Types
Identity	CreateAccount, RotateKey
Recovery	SetSocialRecovery, SetTimelockRecovery, ApproveSocialRecovery, InitiateTimelockRecovery, CancelRecovery, FinalizeRecovery, FinalizeGuardianChange, CancelGuardianChange
Delegation	AuthorizeDerivedKey, RevokeDerivedKey
Threshold	RegisterThresholdKey, ThresholdRecovery
ZK Credentials	AddCredential, RevokeCredential, VerifyCredentialProof, RegisterCircuit, DeprecateCircuit, AnchorAnonymousContent
Capacity	RegisterProvider, SponsorAccount, DeactivateProvider, ClaimProviderUnbonded, MintCapacityCredits, UpdateOraclePrice
Staking	RegisterValidator, DelegateStake, UndelegateStake, Unjail, ClaimUnbonded, SubmitEquivocationEvidence
Social	AnchorGraph, AnchorContent
Messaging	PublishDmKey, SendDm
Schemas	RegisterSchema, SchemaOperation
Indexing	RegisterIndexer, PublishScoreRoot, ChallengeScoreRoot
Moderation	RegisterLabeler, DeactivateLabeler, ApplyLabel, SubscribeLabeler, UnsubscribeLabeler, FileAppeal
Governance	SubmitProposal, CastVote

Table 2: Transaction types by category (49 total).

Time is measured in slots of 600 ms. Epochs group 432,000 slots (≈3 days).

6. Anchored Social Graph Protocol

The Anchored Social Graph Protocol (ASGP) governs how social relationships, content, and messages are stored, anchored, and verified. ASGP separates the social graph into off-chain storage (DA layer) and on-chain anchoring (Merkle roots), providing both scalability and cryptographic verifiability.

The protocol defines five operations: `AnchorGraph` (update edge-set Merkle root), `AnchorContent` (bind content hash to DA reference), `RegisterSchema` (define content/edge types), `PublishDmKey` (register DM encryption key), and `SendDm` (anchor encrypted message).

6.1 Social Vouching and Sybil Resistance

Users create weighted vouch edges to attest to another participant's personhood, stored encrypted on the DA layer and anchored via the ZK graph root. Graph-based Sybil resistance [11] assigns trust scores based on graph topology: accounts with dense connections to the honest core receive high scores, while Sybil clusters receive low scores. ASGP improves on BrightID's model by encrypting vouch edges per-edge and enabling ZK proofs of trust scores without revealing which accounts vouched.

This is probabilistic personhood, not a uniqueness guarantee: small-scale collusion (2–3 accounts with genuine-looking connections to the honest core) remains difficult to detect through graph topology alone. Trust scoring is computed by indexers, not enforced in consensus. To close the trust gap between indexer computation and on-chain verifiability, the protocol supports *staked indexer commitments*: an indexer registers on-chain with a bond, publishes the identifier of a deterministic scoring algorithm (algorithm ID, version, parameters), and posts a Poseidon Merkle root of the computed scores at each epoch. The algorithm must be deterministic given the public vouching topology and a consensus-derived seed (seed = Blake3(epoch‖parent_hash)), eliminating the non-determinism of methods like Louvain. Any party can re-execute the published algorithm over the same input graph and challenge the root if the recomputed result diverges; a successful challenge slashes the indexer's bond and invalidates the epoch root. ZK proofs of trust scores (MerkleInclusion circuit) reference a specific indexer's on-chain root, making the trust chain explicit: *"my score exceeds θ according to indexer I, whose root is on-chain and whose bond backs its correctness."*

Different indexers may register different algorithms; applications choose which indexer's scores to accept, making Sybil resistance a competitive market anchored by staked commitments rather than unverifiable off-chain computation. For use cases where probabilistic Sybil resistance is insufficient, the protocol provides credential-based uniqueness: an external issuer certifies the credential, and the global nullifier set prevents double-use. The social graph and external attestation are complementary tiers, not substitutes: the graph filters most Sybils cheaply, and external attestation provides the hard guarantee when needed.

6.2 Graph Privacy Model

The social graph has two privacy tiers, reflecting the distinct functions of vouching and social connection.

Vouching edges (public topology, encrypted payload). Vouching is a deliberate public act: when account A vouches for account B, the pair (A, B) is published in cleartext on the DA layer. The vouch payload (weight, context, timestamp) is encrypted per-edge. Public vouching topology is necessary for trust scoring: indexers compute bridging coefficients and PageRank-based trust scores (Section 6) over the vouching subgraph at epoch boundaries. This is analogous to PGP key signing or publishing a reference letter: the act of vouching is a public signal, and its value as a trust mechanism depends on observability.

Social edges (fully encrypted). All other edges (follow, block, mute, direct messages) are encrypted independently using ephemeral-static ECIES: a fresh X25519 [9] key pair per edge, Diffie-Hellman with the recipient's static key, domain-separated KDF, and XChaCha20-Poly1305 AEAD. Source, target, edge type, and timestamp are all inside the encrypted payload. Indexers see only the account's Merkle root and DA pointer; they cannot reconstruct the social topology without the recipient's private key. This provides forward secrecy at the edge level: compromising the recipient's long-lived key does not reveal past edges, as ephemeral secrets no longer exist.

ZK bridge between tiers. Verifiers (banks, applications) never see either tier directly. A user proves trust properties ("my vouch score exceeds threshold θ") via the MerkleInclusion ZK circuit against the Poseidon graph root, without revealing who vouched or the underlying topology. The public vouching graph is visible to indexers who compute the scores; the ZK proof hides the graph from the entities that consume the scores.

Metadata side channels. On-chain `AnchorGraph` transactions reveal which account updated its graph root and when, but not the content, targets, or number of edges. Batching multiple edge updates into a single anchor reduces the timing signal. `VerifyCredentialProof` transactions present a stronger metadata concern: the nullifier hides who is proving, but the sponsoring provider is visible through capacity, and the verification timestamp is on-chain. The pattern "provider X verifies N credentials per day" is observable and may constitute business-data leakage for regulated clients. Mitigations include using multiple provider accounts to distribute verification volume and batching proof submissions, but these are operational measures, not protocol-level guarantees.

Vouching topology as correlation skeleton. The public vouch graph can be correlated with `AnchorGraph` timing to partially reconstruct encrypted social edges (e.g. if A and B share a vouch and update roots in close succession).

Batching mitigates but does not eliminate this channel; it is inherent to public-ledger verification with a public vouching topology.

6.3 Content, Schemas, and Portability

Content is stored on the DA layer and anchored on-chain. Schemas are permissionlessly registered with an anti-spam deposit. A signed `PortableProfile` export packages identity, credentials, graph roots, and content references into a verifiable bundle, enabling migration between indexers without loss of provenance.

7. Zero-Knowledge Credentials

ULI applies zero-knowledge proofs *pointwise* on identity attributes, not on the consensus mechanism. The chain operates in the clear; ZK protects specific claims ("I am over 18," "I am a unique human," "I authored this post") without revealing the underlying data.

The credential lifecycle has three phases: (1) **Registration**: the user commits to a credential on-chain via Poseidon commitment; (2) **Proving**: off-chain Groth16 proof generation; (3) **Verification**: on-chain proof check and nullifier recording.

The proof system uses Groth16 [4] on BN254 with constant-size proofs (~128 bytes) and fast verification (3 pairings). In-circuit hashing uses Poseidon [5]. Each proof produces a deterministic nullifier $\nu = \text{Poseidon}(\chi, \tau, s, d)$ that prevents double-use without revealing the prover's identity. The domain separator $d = \text{Poseidon}(\text{scope}, \text{epoch\_window})$ supports *persistent* mode (epoch_window = 0: one-time use, e.g. KYC) and *periodic* mode (epoch_window = e: re-attestation each epoch, e.g. agent sessions, the volume source for break-even burn). χ and τ prevent cross-chain and cross-circuit replay. Periodic nullifiers are pruned after their epoch window expires (default: 2 epochs retention), bounding steady-state storage (see [22]).

Six circuit types are defined:

1. **Personhood** ($\tau = 1$): proves a registered personhood credential without revealing identity.
2. **AttributeRange** ($\tau = 2$): proves $a \geq \theta$ (e.g., age ≥ 18) without revealing the exact value.
3. **MerkleInclusion** ($\tau = 3$): proves membership in a set of up to 2^{16} values.
4. **SelectiveDisclosure** ($\tau = 4$): Merkle inclusion with commitment for anonymous set membership.
5. **AnonymousPost** ($\tau = 5$): proves authorship without revealing the author.
6. **EncryptedDM** ($\tau = 6$): proves a DM was correctly formed for a specific recipient without revealing the sender.

Groth16 requires a per-circuit trusted setup. The plan uses a shared Powers-of-Tau phase 1 ceremony (curve-generic, run once) followed by per-circuit phase 2 ceremonies (six total). This reduces the operational burden from six fully independent ceremonies to one large phase 1 plus six lightweight phase 2 rounds. Security requires at least one honest participant per ceremony. The choice of Groth16 over universal-setup systems (PLONK, STARKs) is a deliberate trade-off: Groth16 produces the smallest proofs (128 bytes) and the fastest verification (3 pairings, ~3–5 ms), which is critical when every validator must verify every proof within a 600 ms slot. PLONK proofs are 2–3× larger and 2–5× slower to verify, directly reducing the per-block ZK throughput ceiling. The trusted setup is therefore a performance optimisation at genesis, not a permanent architectural commitment: circuit versioning via `RegisterCircuit`/`DeprecateCircuit` provides a concrete migration path to transparent proof systems; migration timing will be determined by circuit stability and prover performance benchmarks. Circuit constraints and parameters: see [22], Section 7.

8. Economy

8.1 Capacity Staking

Users interact with ULI without holding tokens. *Providers* (typically application operators) stake tokens to obtain a per-epoch transaction quota (rate limit): $q = s \cdot C_{\text{rate}}$, where $C_{\text{rate}} = 10$ (governance parameter). The quota resets each epoch (non-cumulative) and prevents spam. Transaction *payment* uses Capacity Credits: providers burn ULI to mint CCs at the oracle price, and each transaction deducts from the provider's CC balance (1 CC standard, 10 CC verification; see "Capacity Credits" below). A transaction is admitted iff the provider has remaining quota *and* sufficient CC balance. The minimum provider stake is 1,000 tokens. Sponsorship is enforced at mempool admission and block proposal in addition to execution, so unsponsored transactions cannot occupy block capacity (see the technical specification [22], Section 12).

This eliminates the cold-start problem: application operators bear the cost; end users never interact with the token.

8.2 Validator Staking

Validators register with a self-stake, BLS12-381 public key, and commission rate (in basis points, $\leq$ 10,000). Delegators may stake to any active validator via `DelegateStake`. A validator's voting power equals self-stake plus delegated stake.

Undelegation initiates an unbonding period. Validators may be jailed for equivocation (Section 4). Delegators to a slashed validator lose a proportional share of their delegation, creating incentive to diversify delegations across validators and monitor their behaviour.

8.3 Schema Deposits

Schema registration requires an anti-spam deposit. Any account may register a schema in any namespace by paying the deposit. The deposit remains locked while the schema is active.

8.4 Token Supply and Emission Schedule

The native token (ULI) has a fixed supply of 10^{10} (10 billion). No mechanism creates new tokens beyond the genesis allocation; staking rewards are funded from a pre-allocated emission reserve, not from inflation.

Allocation	ULI	%
Emission reserve	3,500,000,000	35
Ecosystem & grants	2,000,000,000	20
Foundation treasury	1,500,000,000	15
Team & contributors	1,500,000,000	15
Early supporters	1,000,000,000	10
Initial liquidity	500,000,000	5

Table 3: Genesis token distribution.

Team tokens vest over 48 months with a 12-month cliff; early supporter tokens vest over 24 months with a 6-month cliff. The emission reserve is locked in a protocol-controlled account and released only through the epoch reward mechanism.

The per-epoch reward declines geometrically:

$$R(e) = R_0 \cdot \lambda^{\lfloor e/H \rfloor}, \tag{1}$$

where R_0 = 1,000,000 ULI is the initial per-epoch reward, $\lambda = 0.5$ (halving), and $H = 488$ epochs ($\approx$ 4 years at $\approx$ 122 epochs/year).

Year	Per-epoch	Annual	Cumulative
1–4	1,000,000	122M	488M
5–8	500,000	61M	732M
9–12	250,000	30.5M	854M
∞	–	–	976M

Table 4: Emission schedule (approximate).

8.5 Fee Burn

Providers burn ULI to mint Capacity Credits (CCs) at the oracle price; every sponsored transaction then consumes CCs. The aggregate ULI burned by provider P in epoch e is

$$B_P(e) = \sum_{\text{mints in } e} \frac{\text{CC}_{\text{minted}} \cdot P_{\text{CC}}}{P_{\text{ULI}}(e)}, \tag{2}$$

where P_{CC} = \$0.001 is the governance-set CC peg and $P_{\text{ULI}}(e)$ is the oracle price. A verification costs 10 CC (\$0.01); at P_{ULI} = \$0.01 each verification burns 1.0 ULI. At 30% average utilisation of 10,000 capacity units a provider consumes 3,000 CCs per epoch (\$3.00), burning the dollar equivalent in ULI.

Burned tokens are permanently removed from supply. This creates a deflationary force that grows with network activity: more usage → more burn → more replenishment purchases → buy pressure.

Economic model. The capacity stake is functionally a prepaid usage balance: tokens are consumed through usage, not returned upon unstaking. The mechanism adopts Helium's burn-and-mint equilibrium, including fiat-pegged *Capacity Credits* (CCs): providers burn ULI to mint CCs at a fixed dollar price, and transactions consume CCs rather than raw tokens. The term "staking" reflects the mechanism (tokens are locked in a provider account and deducted per epoch) but the economic effect is a per-transaction fee paid in ULI and permanently removed from circulation. Validator staking, by contrast, preserves principal: validators recover their stake upon unbonding.

The mechanical analogy with Helium is exact, but the demand profile differs. Helium's burn depends on voluntary IoT adoption; when usage grew slower than projected, deflationary pressure failed to materialise. ULI's burn is driven by two structural needs, credential verification and AI agent attestation (Section 3.5), that exist independently of any single regulation. Every credential verification, nullifier check, agent attestation, and account recovery consumes capacity and burns tokens. The proliferation of AI agents that transact on behalf of humans creates additional demand: every delegation chain requires a credential-backed personhood anchor. Regulatory trends favouring digital identity infrastructure are a tailwind, but the demand thesis does not depend on specific mandates or their timing.

Capacity Credits. A Capacity Credit (CC) is a non-transferable, non-tradeable prepaid usage unit pegged at **$0.001/CC** (governance-set, fixed fiat). Providers burn ULI to mint CCs at the prevailing oracle price: $\text{CC}_{\text{minted}} = \text{ULI}_{\text{burned}} \times P_{\text{ULI}}/P_{\text{CC}}$, where $P_{\text{CC}} = \$0.001$. A standard operation consumes **1 CC** ($0.001); a verification (`VerifyCredentialProof`) consumes **10 CC** ($0.01). This produces fiat-predictable provider costs (Table 5), dollar-denominated token demand, and preserved deflationary burn (ULI is permanently destroyed when CCs are minted). A stabilising feedback loop results: low token price → more ULI burned per CC → stronger deflation; high price → less burn → moderated deflation. The oracle computes a *stake-weighted median* at each epoch boundary (≥3 validators, 20% deviation cap, last-valid-price fallback; see [22]). CC minting uses a TWAP (time-weighted average price) computed over a rolling 7-epoch window, smoothing short-term volatility. *Bootstrap mode*: before exchange listing, the Foundation sets P_{ULI} via governance, transitioning to the validator-driven median when the active set is sufficiently decentralised.

Designed asymmetry. The cost asymmetry in Table 5 is intentional, not a weakness. Social transactions (posts, follows, likes) are batched and nearly free; this is the acquisition channel that builds the graph and onboards users. Identity-verification transactions (ZK proofs, nullifier checks) burn at full rate; this is the monetization channel that sustains the security budget. The model is freemium: cheap adoption drives graph growth, and high-value verifications fund the network. If social burn were high, adoption would stall; if verification burn were low, the token would lack utility demand. The throughput ceiling for ZK-heavy blocks is a capacity constraint, not a design flaw. At steady state (≤5% ZK mix), the constraint is not binding. Beyond parameter tuning (block size, slot duration), the primary scaling mechanism is *recursive proof aggregation*: an off-chain aggregator combines N ZK proofs into a single recursive proof verified once per block. Efficient recursion over Groth16/BN254 requires non-native field arithmetic (cycle-of-curves or folding schemes); this is a primary motivation for the PLONK/STARK migration path described in Section 7. Recursive aggregation is expected to precede rather than follow proof-system migration, as the scaling bottleneck manifests before circuit stability concerns justify the migration cost. The circuit versioning mechanism (`RegisterCircuit`/`DeprecateCircuit`) is designed to support this transition without chain halt.

Three downside scenarios remain plausible: (1) demand grows slower than projected (fewer enterprises adopt credential verification infrastructure); (2) incumbents capture the verification market with proprietary infrastructure; (3) a competing open protocol reaches critical mass first. In each scenario, fee-burn volume is lower, weakening the deflationary force and reducing the security budget (Section 8.8 quantifies the break-even activity level). The emission reserve (3.5B ULI) permanently exceeds total uncapped emission (976M ULI), providing an indefinite runway for adoption regardless of timing.

Who buys ULI and why. Four mechanisms create demand for ULI:

1. **Fee burn replenishment** (primary at scale). Providers must continuously purchase ULI to replace burned stake.
2. **Validator self-stake.** Minimum 10,000 ULI; validators stake above minimum to increase reward share.
3. **Delegation.** Token holders delegate to validators for yield (net of commission).
4. **Deposits.** Labeler deposits (10,000 ULI, refundable) and schema deposits (non-refundable, permanently removed from supply).

The demand profile is institutional: application operators, validators, and infrastructure providers, not retail users. End users never purchase, hold, or interact with ULI.

Provider as middleware. Relying parties that prefer not to manage blockchain infrastructure interact with ULI through a provider's API: the provider handles staking, burn replenishment, and key management internally; the relying party sees a standard REST endpoint that returns a credential verification result. The token is infrastructure plumbing, invisible to the end customer and to the relying party.

Who holds ULI. All long-term holders have functional reasons: validators stake for security and yield, providers maintain working capital for burn replenishment, delegators earn commission-net yield by backing validators, and the foundation stakes for network security (Section 8.8). There is no speculative holding thesis; token value is a function of network usage, not narrative.

Provider unit economics. Table 5 shows the annual cost for an application provider under CC pricing. Because CCs are fiat-pegged ($0.001/CC), provider costs are fixed in dollar terms regardless of token price.

Tiered CC pricing. Verification transactions (`VerifyCredentialProof`) consume **10 CC** ($0.01) each, reflecting their higher computational cost (ZK proof verification) and economic value. All other transaction types consume **1 CC** ($0.001).

Identity-infrastructure providers (rows 1–2) submit individual verification transactions at 10 CC each. *Consumer-social* providers (rows 3–4) batch user actions into `AnchorGraph` transactions at 1 CC each (one per 1,000-user shard per epoch), reducing consumption by orders of magnitude.

Mode	Users	Tx/epoch	CC/epoch	Annual cost (USD)
Identity infrastructure (10 CC per verification)				
	10K	100K	1M	$122K
	100K	1M	10M	$1.22M
Consumer social (batched via `AnchorGraph`*, 1 CC)*				
	10K	10	10	$1.22
	100K	100	100	$12.2

Table 5: Provider annual cost under Capacity Credit pricing (1 CC = $0.001, fiat-pegged; cost independent of token price). Annual cost = CC/epoch × $0.001 × 122 epochs/year. Consumer-social rows assume 1 `AnchorGraph` per 1,000 users per epoch, batching all user actions into a single hash-anchor.

8.6 Reward Schedule and Validator APR

Each epoch distributes $R(e)$ ULI (Equation 1) among active validators proportional to their effective stake. To prevent excessive APR when staking participation is low, the protocol scales the effective emission with a participation factor:

$$R_{\text{eff}}(e) = R(e) \cdot \min\left(1,\ \frac{S_{\text{total}}}{S_{\text{target}}}\right), \qquad S_{\text{target}} = 0.25 \times S_{\text{max}} = 2.5\text{B ULI}. \tag{3}$$

When fewer tokens are staked than the target floor, emission scales down proportionally; unissued rewards remain in reserve, extending the emission schedule. The annualised reward rate is therefore:

$$\text{APR} = \frac{R_{\text{eff}}(e) \cdot 122}{S_{\text{total}}}. \tag{4}$$

Staked (% of supply)	S_{total}	R_{eff}/epoch	Year-1 APR
5%	500M	200,000	4.9%
10%	1.0B	400,000	4.9%
25%	2.5B	1,000,000	4.9%
50%	5.0B	1,000,000	2.4%

Table 6: Year-1 APR under emission scaling (R_0 = 1,000,000, S_{target} = 2.5B). APR is capped at ≈ 4.9% regardless of how low staking participation falls. The cap operates by reducing absolute emission, not by redistributing it; unissued tokens remain in reserve.

Reserve sufficiency. The uncapped geometric emission series sums to ≈ 976M ULI, well below the 3.5B reserve. The reserve is never exhausted under any staking scenario: the surplus of 2.524B ULI remains available for governance-directed purposes. Emission scaling further conserves the reserve: whenever staking participation is below S_{target}, unissued rewards remain in reserve, extending the schedule indefinitely.

8.7 Slashing Economics

Equivocation slashing removes 10% of the offending validator's stake. Slashed tokens are permanently burned, not redistributed. The validator is jailed for a minimum of 7 epochs (≈ 21 days). Delegators to a slashed validator lose a proportional share.

8.8 Security Budget and Terminal Economics

The minimum cost to compromise consensus safety is the cost of acquiring $> 2S/3$ stake. In practice, the acquisition cost far exceeds the spot-price calculation: purchasing a supermajority on the open market would exhaust available liquidity and drive the price up by orders of magnitude before completion. Table 7 provides a lower-bound estimate (spot price × required stake, ignoring market impact) at three staking levels and three token prices.

Staked	S_{total}	**>2S/3 req.**	**Min. acquisition cost (USD)**		
			\$0.01	\$0.10	\$1.00
5%	500M	334M	\$3.3M	\$33M	\$334M
25%	2.5B	1.67B	\$16.7M	\$167M	\$1.67B
50%	5.0B	3.34B	\$33.4M	\$334M	\$3.34B

Table 7: Lower-bound cost of consensus attack ($> 2S/3$ stake at spot price, ignoring market impact, slippage, and slashing). Real acquisition cost is substantially higher due to thin float and vesting locks.

Early-network security. At genesis, the foundation stakes its entire allocation (1.5B ULI, 15% of supply) to validators, establishing a minimum staking floor independent of market participation. Combined with team tokens (15%, vesting), emission reserve (35%, locked), and ecosystem grants (20%, capped at 0.5% of supply per quarter), the year-one free float is ≤15% (early supporters + liquidity), rising to at most ∼17% if grants are fully disbursed. All locks and caps are consensus-enforced. Note: during the permissioned phase, the Foundation controls sufficient stake to pass governance votes unilaterally; grant caps are reputational self-constraints until permissionless transition, not cryptoeconomic guarantees. The initial validator set is permissioned (progressive decentralisation). Year 1–3 security is funded by foundation stake and vesting locks, not by market demand, a bootstrap condition shared by all new PoS chains but not disguised here by analogy with mature networks. Market-driven security activates when $S_{\text{total}} > S_{\text{target}}$. Progressive decentralisation follows three verifiable milestones: (1) at 10% staked, the validator set opens to any operator posting a bond $\geq B_{\min}$ and demonstrating state synchronisation; (2) at S_{target} (25%), full permissionless entry; (3) at epoch E_{fallback} (governance-set, default 366 ≈ 3 years), permissionless entry activates unconditionally regardless of staking level; the permissioned phase is bounded even if adoption is slower than projected.

Break-even activity level. With $\alpha = 0.5$ active from genesis, validators earn $R_{\text{eff}} + \alpha \times B_{\text{total}}$ per epoch and $(1-\alpha) \times B_{\text{total}}$ is permanently burned. The token is deflationary when $(1-\alpha) \times B_{\text{total}} \geq R_{\text{eff}}$, i.e. $B_{\text{total}} \geq 2 \times R_{\text{eff}}$. At year-1 parameters ($R_0$ = 1M, 25% staked, R_{eff} = 1M/epoch), break-even requires $B_{\text{total}} \geq$ 2M ULI burned per epoch. Each verification consumes 10 CC (\$0.01), burning $\$0.01/P_{\text{ULI}}$ of ULI. At the reference price $P_{\text{ULI}} = \$0.01$, this is 1.0 ULI per verification, so 2M verifications per epoch ≈ 7.7 sustained TPS. Because CCs are fiat-pegged, break-even TPS scales with price: $\text{TPS}_{\text{BE}} = 7.7 \times P_{\text{ULI}}/\$0.01$. This is a stabilising feedback loop: low price → lower break-even *and* more ULI burned per CC → stronger deflation when most needed. Below break-even the token is inflationary; above it, deflationary.

Annual verification perspective. The per-epoch framing above assumes 10 transactions per user per epoch, appropriate for high-frequency integrations (automated credential refresh, AI agent re-attestation per session). For lower-frequency use cases such as KYC onboarding, the break-even expressed in annual verifications depends on staking participation:

Staked	Break-even	Annual tx	Users (2/yr)	Users (5/yr)
5%	1.5 TPS	48.6M	24.3M	9.7M
10%	3.1 TPS	97.4M	48.7M	19.5M
25%	7.7 TPS	243M	122M	48.7M

Table 8: Break-even annual verifications and required user base at two verification frequencies.

Break-even TPS and the security budget grow together with staking participation (Figure 3). Low staking makes break-even reachable but reduces the cost of consensus attack (Table 7); high staking increases security while break-even remains modest.

Three levers exist if low-frequency verification dominates: (1) agent attestation traffic: AI agents re-attesting per session generate $\gg 2$ transactions per entity per year, and delegation chain operations (Section 3.3) count as burn-generating transactions, bridging the gap between KYC-frequency and the per-epoch model; (2) governance may increase the CC cost per verification (currently 10 CC), increasing ULI burn per transaction at the cost of higher provider fees, subject to provider demand elasticity; (3) emission scaling (Equation 3) prevents excessive dilution at low participation.

Cold-start path. The 7.7-TPS break-even assumes 25% staking participation. At launch, staking participation is low and emission scaling (Equation 3) reduces both the emission *and* the break-even proportionally. At 5% staked, break-even is 1.5 TPS (≈133K verifications/day); at 15%, it is 4.6 TPS (≈400K verifications/day).

Sensitivity analysis. Figure 3 combines both thresholds. At low staking, emission scaling reduces the break-even TPS (1.5 at 5% vs. 7.7 at 25%) but increases the token price required for adequate security. At high staking, security is cheap but the break-even TPS rises. The network is in the safe zone when operating above *both* curves; below that, emission reserves provide runway.

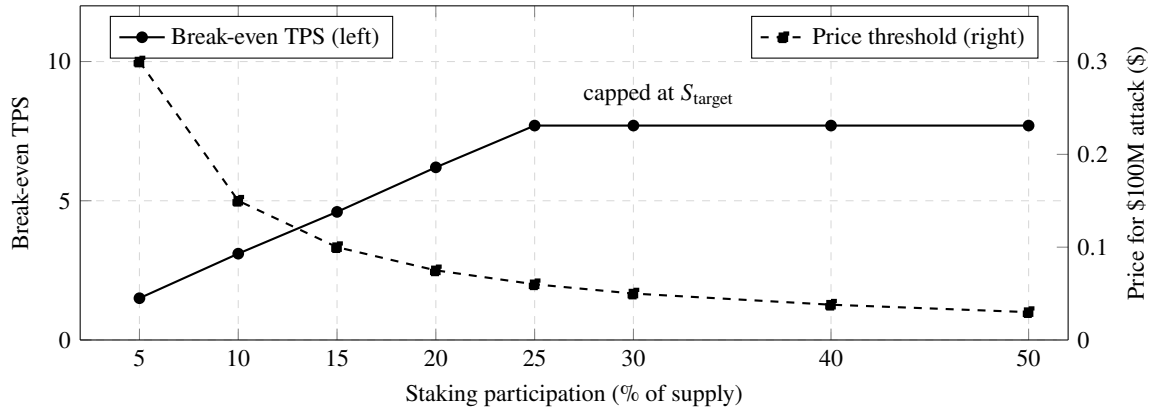

Figure 3: Security and inflation thresholds vs. staking participation (year 1). Solid (left axis): sustained identity-verification TPS for burn to equal emission; above the curve the token is deflationary. Dashed (right axis): token price at which a $> 2S/3$ attack costs $100M (spot, ignoring slippage). The safe zone is the region above both curves.

Terminal state. A governance-set parameter $\alpha \in [0, 1]$ ($\alpha = 0.5$ from genesis) splits each capacity burn: fraction α is distributed to active validators proportional to their stake; fraction $(1 - \alpha)$ is permanently burned. At $\alpha = 0.5$, half of all capacity fees fund validator revenue and half sustain deflationary pressure. Validators therefore earn $R_{\text{eff}} + \alpha \times B_{\text{total}}$ per epoch from day one; the fee-derived component is not a future mechanism activated upon reserve exhaustion but an immediate supplement to emission rewards.

As emission halves geometrically, fee revenue becomes the dominant validator income; there is no reserve-exhaustion cliff.

9. Moderation

Content stored on the DA layer is uncensorable at the protocol level. Moderation operates through a *labeler* system that separates the act of labelling from the act of filtering.

Any account may become a labeler by registering with a 10,000-token deposit. Labelers publish labels on content or accounts; users subscribe to labelers they trust (up to 64 subscriptions); client applications filter content based on subscribed labels. The filtering logic is entirely client-side. On-chain appeals provide a transparency mechanism for dispute resolution.

Why opaque ranking becomes structurally difficult. ULI breaks the coupling between identity, social graph, and ranking: (1) identity is protocol-owned, not platform-owned, so switching indexers does not require re-creating an account; (2) the social graph is user-owned and auditable via on-chain Merkle roots; (3) ranking is a competitive market where multiple indexers compete on algorithm quality, and users switch at zero cost. This makes manipulative ranking detectable, switchable, and competitively disadvantaged.

Formal specifications. The companion yellowpaper [22] provides formal algorithm specifications with complexity analysis (Section 10), performance bounds with empirical benchmarks and per-operation cost tables (Section 11), and a structured security analysis including threat model and catastrophic recovery scenarios (Section 12).

10. Related Work

Table 9 compares ULI with existing decentralised social protocols across ten dimensions.

	ULI	**AT Proto** [19]	**DeSo** [7]	**DSNP** [8]	**Farcaster** [16]	**Nostr** [17]
Identity	Numeric ID, DID	did:plc	Public key	DSNP Id (u64)	ETH FID	Public key
Key rot.	Yes (16 ret.)	Yes (PLC op)	No	Yes	Yes (L1)	No
Recovery	Social, TL, FROST	PLC rotation	None	Provider	ETH-based	None
Content	Off-chain (DA)	Off-chain (PDS)	All on-chain	Off-chain	Off-chain	Relays
ZK cred.	6 circuits	None	None	None	None	None
Encrypt.	Per-edge AEAD	None	None	None	None	NIP-44
Agent attest.	Derived keys	App passwords	None	None	None	None
Token req.	No (capacity)	No	Yes	Yes	Yes (ETH)	No
Moderation	Labelers	Labelers	None	Provider	Channel	Relay
Graph priv.	Layered[1]	Public	Public	Public	Public	Public

Table 9: Comparison of decentralised social identity protocols.

Proof-of-personhood and attestation systems. Worldcoin [12] achieves uniqueness through iris biometry captured by a proprietary hardware device (the Orb). The iris is an immutable biometric identifier: unlike a cryptographic key, it cannot be rotated upon compromise. The system concentrates credential issuance in a single hardware manufacturer, creating a trust dependency and a data-protection surface that constrain deployment in jurisdictions with strict consent requirements. ULI addresses a different architectural point: it treats external attestation (biometric or otherwise) as an optional credential layer rather than a protocol prerequisite, providing graph-based Sybil resistance as a verifiable trust-scoring market (staked indexers with fraud proofs) and making strong external credentials portable and privacy-preserving via ZK proofs. Even in jurisdictions where biometric approaches are legal, the credential-agnostic model offers advantages: no specialised hardware, no central issuer, and revocable credentials.

Gitcoin Passport aggregates attestation stamps from external services into a composite personhood score. Because several stamp categories correlate with accessible on-chain or social activity, the composite score may not reliably distinguish unique individuals from determined attackers.

[1] Social edges encrypted (per-edge AEAD); vouching topology intentionally public for trust scoring. See Section 6.2.

Decentralised social protocols. AT Protocol [19] (Bluesky) has demonstrated at scale that portable identity drives social adoption, and it did so without a blockchain or token. This raises two direct questions: why does ULI require a sovereign L1, and why a native token?

Why a canonical state machine. The blockchain is not for the bank; it is for the user. Institutions interact through a provider's REST API and never see the consensus layer (Section 8.5). The W3C/OID4VP bridge makes ULI credentials consumable by any standards-compliant verifier, but the bridge does not eliminate the chain. It makes the chain's *output* portable; the chain remains the irreducible trust layer that *produces* that output. Without it, there is no neutral party to (1) maintain the global nullifier set, (2) resolve key-recovery conflicts, or (3) anchor credential commitments to a tamper-evident log.

A centralised database satisfies none of these: the operator can censor nullifiers, reverse recoveries, and alter commitments under jurisdictional pressure. A federated database partitions the nullifier set, destroying global uniqueness, the property that makes portable credentials meaningful.

Neither operation can be added to AT Protocol without introducing a consensus mechanism, at which point AT Protocol becomes a blockchain.

Why a sovereign L1 (not a rollup or existing chain). The case for a sovereign L1 rests on two arguments with different strengths, and intellectual honesty requires distinguishing them.

(a) Identity state alone does not require a sovereign chain. The nullifier set, key-recovery log, and credential commitments are low-frequency, low-volume state (kilobytes per epoch). This state could technically reside on an Ethereum L2 at low cost. A rollup would inherit the host chain's security while sacrificing governance independence: the host chain's upgrade schedule, fee market, and censorship policy are decided by a community whose priorities are not identity infrastructure. For identity, this is not an abstract risk. A nullifier set records "this person has already been verified," a censorship-sensitive state. If a regulator pressures the host chain's foundation or sequencer operator, the nullifier set inherits that jurisdictional exposure. Sovereign consensus provides governance isolation: the identity layer's censorship policy is set by its own validator set, not delegated to a general-purpose chain.

(b) Encrypted social graph data requires sovereign DA. The social graph (per-edge encrypted edges, DMs, content anchors) is high-volume data (gigabytes at scale). Ethereum blobspace costs \$1–10 per MB (EIP-4844); an encrypted social graph for millions of users would cost orders of magnitude more than the identity-verification revenue it generates. Sovereign data availability eliminates this cost mismatch.

(a) + (b) together justify a sovereign L1. Running identity state on an external L2 and social-graph DA on a separate sovereign layer creates operational complexity without benefit: two validator sets, two fee markets, two governance structures for what is architecturally a single system. Co-locating both on a sovereign L1 with a unified validator set provides operational independence, jurisdictional diversity, and cost-effective DA in a single deployment.

This justification is conditional: it depends on social applications building the encrypted graph. If only the identity-verification stream materialises, argument (b) weakens and the case for a sovereign L1 reduces to governance independence, a defensible but narrower position. If neither condition holds (social adoption does not materialise and credential issuers do not value governance independence), the protocol's identity primitives (nullifier set, ZK circuits, recovery mechanism) could operate on an existing L2 at lower infrastructure cost. The architecture supports this graceful degradation: the ZK circuits and credential format are chain-agnostic, and the W3C/OID4VP bridge already abstracts the consensus layer from relying parties. The team's bet is that both conditions are met for the target market; the sovereign L1 is a deliberate investment in jurisdictional neutrality and cost-effective social-graph DA, not a prerequisite for the identity primitives themselves.

Why a native token (not stablecoin fees). Stablecoin issuers (Circle, Tether) can freeze addresses and blacklist contracts. Identity infrastructure cannot depend on an external party with a kill switch over its payment rail. A native token has no issuer who can unilaterally halt the system. Second, validators paid in stablecoins have no economic alignment with protocol health; a native token ties validator wealth to the network they secure. Third, bootstrapping a validator set requires funding security from genesis through emission reserves; this is not possible with an external asset the protocol does not control.

Why BFT machinery at genesis (not later). The initial validator set is permissioned (Section 8.8): year one, the network is functionally a consortium under foundation stewardship, and the critique that "a federated database can be pressured by a jurisdiction" applies equally to this phase. The difference is architectural: the protocol, state format, and economic model are identical in both phases; only the validator admission policy changes. Retrofitting consensus onto a running federated system requires re-engineering the state model, re-deploying all clients, and migrating all data; no production system has done this successfully. The BFT machinery is deployed from genesis so that permissionless entry is a configuration change, not a re-architecture. Progressive decentralisation has a public, verifiable exit condition: permissionless entry is enabled when $S_{\text{total}} > S_{\text{target}}$, or unconditionally at epoch E_{fallback} (default $366 \approx 3$ years), ensuring the permissioned phase is bounded regardless of adoption trajectory.

ULI therefore addresses a different set of requirements than AT Protocol. Social applications that need

protocol-level recovery and ZK credential verification require capabilities that AT Protocol's architecture does not provide.

DeSo [7] is architecturally closest, sharing Fast-HotStuff consensus and derived-key delegation. However, DeSo stores all content on-chain and provides no privacy primitives.

DSNP/Frequency [8] separates content from consensus but requires governance referenda for schema registration and provides no ZK capabilities.

Farcaster [16] achieves decentralised identity via Ethereum but requires L1 gas for key rotation and provides no ZK credentials or protocol-level encryption.

Nostr [17] is the most minimal: identity is a secp256k1 key with no rotation, recovery, delegation, or protocol-level encryption beyond NIP-44.

BrightID [11] pioneered graph-based Sybil resistance but its graph is publicly visible and its trust algorithms remain vulnerable to small-scale collusion. ASGP adopts the vouching model but adds *staked indexer commitments*: indexers register deterministic scoring algorithms on-chain with a bond, publish Poseidon Merkle roots of scores each epoch, and face slashing if any party re-executes the algorithm and obtains a different result. Verifiers see only a ZK proof of the score against a bonded on-chain root, not the graph topology itself. This transforms Sybil scoring from an unverifiable black box into a competitive market with economic accountability, a property absent from all existing personhood systems.

	Worldcoin	**BrightID**	**Gitcoin Passport**	**ASGP**
Method	Iris biometry	Public graph	Stamp aggregation	Graph topology
Scoring input	Binary (scan)	Public graph	Public stamps	Public vouch topology*
Verifier sees	Iris hash	Full graph	Composite score	ZK proof of score
Score verifiab.	Orb trust	None	None	Fraud-provable, bonded
Portability	Orb-bound	Limited	Platform-bound	Cross-platform
Hardware	Orb (proprietary)	None	None	None
Revocable	No (iris)	Yes	Yes	Yes

*Vouch topology (who vouched whom) is public for scoring; vouch payloads (weight, context) and all non-vouch edges (follows, blocks, DMs) are encrypted per-edge.

Table 10: Personhood and trust-scoring system comparison.

Existing protocols address identity, social graph, and credentials separately; these partial solutions do not compose into a unified system. ULI integrates them under a single identity substrate, combining purpose-built chain architecture, content-consensus separation, composable moderation, graph-based trust scoring, and ZK credentials, with a token-free user experience via capacity staking.

Operation	**ULI**	**AT Proto**	**Farcaster**	**DeSo**	**Nostr**
Account	1 cap.	Free	Hub fee	DeSo tx	Free
Post	1 cap.	Free	Hub stor.	DeSo tx	Relay
Recovery	On-chain	PLC log	ETH-based	None	None
ZK cred.	On-chain	N/A	N/A	N/A	N/A
Encryption	Per-edge	None	None	None	NIP-44
Agent auth.	Derived key	App pwd	N/A	N/A	N/A

Table 11: Operation cost comparison across protocols.

11. Conclusion

ULI provides key-independent recoverable identity, zero-knowledge credential verification, an encrypted social graph with ZK attestation, and cryptographic agent authorization, without requiring end users to hold tokens.

Three technical properties distinguish the protocol from existing systems:

1. **Recovery and key sovereignty at the protocol level.** ULI offers three complementary mechanisms (social, timelock, FROST threshold), ensuring that key compromise triggers recovery, not identity loss. NEAR Protocol's named accounts with function-call keys and ERC-4337 account-abstraction wallets address the

same pain at different layers: NEAR provides protocol-level key rotation but not identity-specific recovery or ZK credentials; AA wallets provide contract-level social recovery but per-wallet, not standardised across all accounts. ULI's contribution is a uniform recovery model integrated with the identity, credential, and delegation system, not recovery as an isolated feature.

2. **Personhood without irrevocable biometrics.** Graph-based personhood scoring via social vouching and optional external attestation, verifiable through ZK proofs. The guarantee is probabilistic: graph topology detects isolated Sybil clusters but not small-scale collusion with genuine-looking connections. In jurisdictions where biometric collection is regulated or prohibited, this trade-off (probabilistic scoring vs. biometric uniqueness) offers an alternative path to personhood verification.
3. **Verifiable trust scoring with privacy at the verifier.** Staked indexers compute trust scores over public vouching topology and commit fraud-provable Merkle roots on-chain. Verifiers see only a ZK proof that a score exceeds a threshold against a bonded indexer's root, not the graph, not the individual vouches. This is the property that applications requiring data minimisation need and that public-graph protocols (AT Protocol, BrightID, Farcaster, Nostr) cannot provide: verifiable Sybil resistance without graph access at the point of verification.

Several design decisions remain open or have been recently resolved. On-chain governance is implemented with a proposal lifecycle (submit with deposit, stake-weighted voting with 33% quorum and 67% threshold, epoch-boundary tally, enact after delay) supporting three actions: `SetParameter`, `ScheduleUpgrade`, and `EmergencyValidatorReset`. The proof system requires one shared Powers-of-Tau phase 1 plus six per-circuit phase 2 Groth16 ceremonies; migration to a transparent scheme (PLONK or STARKs) via `RegisterCircuit` and `DeprecateCircuit` is targeted for mainnet launch or immediately after, depending on circuit stability. The fee burn mechanism creates a direct link between network usage and token demand; its parameters (CC cost per operation, C_{rate}) are governance-adjustable to adapt to changing market conditions. W3C VC and OID4VP compatibility (Section 3.5) enables integration with enterprise identity pipelines without requiring those systems to adopt ULI-native formats.

References

[1] M. Yin, D. Malkhi, M. K. Reiter, G. Golan-Gueta, and I. Abraham. HotStuff: BFT consensus with linearity and responsiveness. In *Proc. ACM PODC*, pages 347–356, 2019.

[2] M. M. Jalalzai, J. Niu, C. Feng, and F. Gai. Fast-HotStuff: A fast and resilient HotStuff protocol. *IEEE Trans. Dependable and Secure Computing*, 21(4):2478–2493, 2023.

[3] D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. In *Advances in Cryptology, ASIACRYPT 2001*, pages 514–532. Springer, 2001.

[4] J. Groth. On the size of pairing-based non-interactive arguments. In *Advances in Cryptology, EUROCRYPT 2016*, pages 305–326. Springer, 2016.

[5] L. Grassi, D. Kales, R. Khovratovich, A. Roy, C. Rechberger, and M. Schofnegger. Poseidon: A new hash function for zero-knowledge proof systems. In *USENIX Security Symposium*, 2021.

[6] C. Komlo and I. Goldberg. FROST: Flexible round-optimized Schnorr threshold signatures. In *Selected Areas in Cryptography (SAC)*, pages 34–65. Springer, 2020.

[7] DeSo Foundation. DeSo: The decentralized social blockchain. White paper, 2022.

[8] Unfinished Labs. DSNP: Decentralized social networking protocol specification. Version 1.2, 2023.

[9] D. J. Bernstein. Curve25519: New Diffie-Hellman speed records. In *Public Key Cryptography, PKC 2006*, pages 207–228. Springer, 2006.

[10] M. Hamburg. Decaf: Eliminating cofactors through point compression. In *Advances in Cryptology, CRYPTO 2015*, pages 705–723. Springer, 2015.

[11] S. Shahaf, E. Duffy, S. Kalodner, and M. Naor. Who watches the watchmen? A review of subjective approaches for Sybil-resistance in proof of personhood protocols. *Frontiers in Blockchain*, 2020.

[12] V. Buterin. What do I think about biometric proof of personhood? Technical note, 2023.

[13] E. Pariser. *The Filter Bubble: What the Internet Is Hiding from You*. Penguin Press, 2011.

[14] C. A. Bail, L. P. Argyle, T. W. Brown, et al. Exposure to opposing views on social media can increase political polarization. *Proc. Nat. Acad. Sci.*, 115(37):9216–9221, 2018.

[15] F. P. Santos, Y. Lelkes, and S. A. Levin. Link recommendation algorithms and dynamics of polarization in online social networks. *Proc. Nat. Acad. Sci.*, 118(50), 2021.

[16] Farcaster Protocol. Farcaster protocol specification. 2023.

[17] Nostr Contributors. Nostr: Notes and other stuff transmitted by relays. NIP-01, 2023.

[18] P. Wuille. BIP-340: Schnorr signatures for secp256k1. Bitcoin Improvement Proposal, 2020.

[19] Bluesky PBC. AT Protocol: The social internet. Protocol specification, 2024.

[20] W3C. Decentralized identifiers (DIDs) v1.0. W3C Recommendation, 2022.

[21] W3C. Verifiable credentials data model v2.0. W3C Recommendation, 2024.

[22] ULI. ULI: Technical specification. Yellowpaper, 2026.