AetherIdentity: A Falsification-First Investigation of Decentralized Agent Identity in the Aether Family

A Falsification-First Investigation of Identity Proof, Continuity, Reputation, Sybil Resistance, and Delegation in Multi-Agent Software Ecosystems

Samir Baladi

Interdisciplinary AI Research — Ronin Institute / Rite of Renaissance

Research Article — Distributed Systems / Security / Multi-Agent Systems
DOI: https://doi.org/10.5281/zenodo.24102739
ORCID: 0009-0003-8903-0029 · gitdeeper@gmail.com
Record ID: AETHERIDENTITY-01 · Version 0.1.0 — November 2026
License: MIT (code) + CC-BY-4.0 (documentation)

Status: *Formal Architecture Proposal — no implementation claimed. AetherIdentity is an architectural hypothesis under investigation, not a committed project. It is the provisional eighth dimension of the Aether family. No implementation, test, or experiment is reported. No real-world deployment has been evaluated. The final project structure is undetermined.*

Abstract

The Aether family comprises seven preprints on how agent-generated software changes are stored, integrated and verified, specified and evidenced, synchronized, authorized, monitored, and orchestrated. All of them rely on a notion of *who* acts, but none treats identity itself: how an agent proves who it is, how that identity persists across restarts and key changes, how its behavioral history is attached to it, how many identities controlled by one entity are handled, and how authority is passed on with bounded scope. This paper investigates whether those five concerns amount to a genuine gap that would justify an eighth dimension, tentatively AetherIdentity, or are an artifact of how the existing dimensions were scoped. The method is falsification-first. Five failure conditions (F1–F5), each of which would show the candidate does not deserve to be a separate dimension, and six requirements (R1–R6) are stated before the candidate is described. The candidate is then specified as five layers (identity, continuity, reputation, Sybil resistance, delegation), and two propositions are stated with proof sketches. The first (P1) restates determinism of local identity issuance: identifier derivation is a deterministic function of a public key and a label, identity admission is a deterministic function of a record and its causal closure, and local Sybil evidence enters only as signed claims with a deterministic effect rule, so AetherMesh's admission-determinism assumption A6 is preserved. The second (P2) states that any decentralized identity protocol leaves an irreducible trust anchor, an initial set of identities trusted without protocol-level proof, because admission chains are finite and well-founded. By argument alone, F1, F2, and F3 are not falsified, each with a stated reservation; F3 is the most fragile, since delegation overlaps strongly with AetherGovern's authority chains. F4 (decentralizability) is not falsified with residuals: the trust anchor, Sybil resistance beyond cost or attestation, and recovery after total key loss cannot be removed. F5 (observable benefit) is untested. Nine metrics, five baselines, six scenarios, and an evaluation protocol are proposed but not executed. AetherIdentity is a candidate, not a result. No implementation, experiment, or measurement is reported.

Keywords: *Decentralized Identity; Self-Certifying Identifiers; Key Rotation; Reputation; Sybil Resistance; Delegation; Capabilities; Trust Anchor; Falsification; Agentic Software Engineering; AetherIdentity.*

Evidentiary status: *hypothesis under investigation. Every statement is labeled as a definition, a hypothesis, an observation, a proof sketch, or a formula output. Propositions P1–P2 are proof sketches, not machine-checked. No empirical result is reported. Descriptions of third-party work follow the cited sources and were not independently verified.*

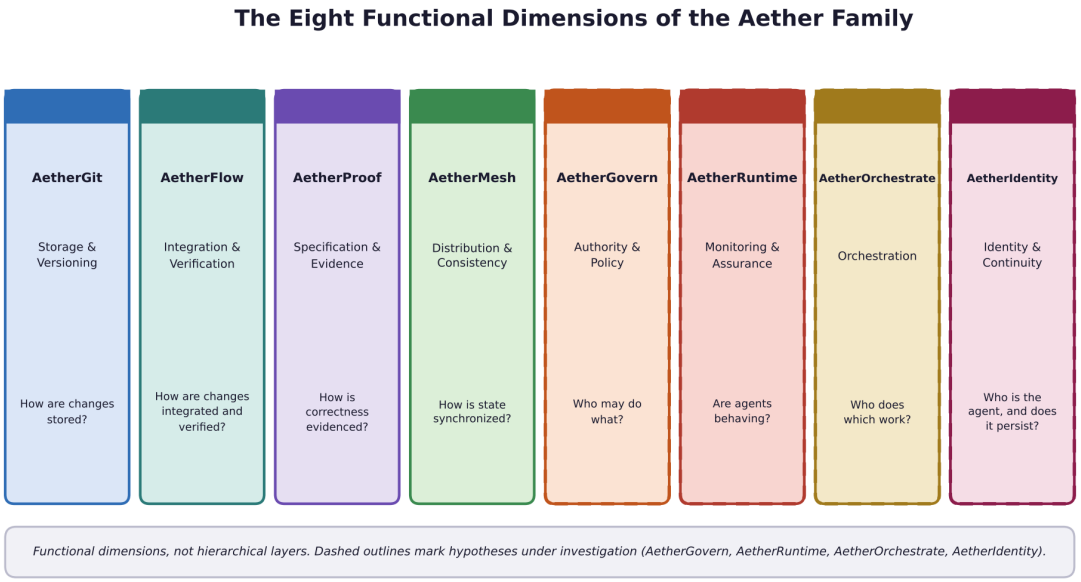


Figure 1 (Fig1_Eight_Dimensions.png). The eight functional dimensions of the Aether family. AetherGovern, AetherRuntime, AetherOrchestrate, and AetherIdentity are hypotheses under investigation, not confirmed dimensions. The dimensions are functional, not hierarchical. Redrawn for this record.

Table of Contents

1. Introduction	3
2. Background and Related Work	5
3. Formal Framework	7
4. Failure Conditions F1–F5	11
5. Requirements R1–R6	14
6. Architecture: The Five Layers	15
7. Interfaces with the Aether Family	21
8. Evaluation Design (Proposed)	23
9. Threats to Validity	25
10. Open Questions Q1–Q20	25
11. Conclusion	27
References	29
Research Status Statement	30
Appendix A — Notation	30
Appendix B — Derivations	30
Reconciliation Notes	32

1. Introduction

1.1 The Identity Gap at Agent Scale

Every cooperative system needs some answer to the question *who is acting?* Human institutions answer it with names, documents, institutions that vouch for them, and a shared history. For software agents the answer is usually a cryptographic key: an agent signs what it does, and others check the signature. A key is a good start, but it is only a start. It does not say whether the agent that holds the key today is the one that held it yesterday, whether it has behaved well, whether it is one agent among many controlled by the same operator, or what it is allowed to pass on to others.

When agents are few and administered by one organization, these questions are settled administratively. At agent scale, with many agents created and retired by many parties and no single registry, they become design questions. How does an agent prove it is who it claims, without a certificate authority that every participant must trust? How does its identity survive a restart, a migration to another host, or a key rotation? How is its record of past behavior attached to it, so that others can rely on that record? How is a flood of fabricated identities handled? And how does an agent hand on part of its authority to another agent for a bounded purpose?

The identity gap is the observation that these five questions are asked implicitly by every layer of a decentralized agent ecosystem, and answered explicitly by none. This paper does not argue that the gap is a genuine dimension of the design. It asks whether it is, and states in advance what evidence would show that it is not.

1.2 Relationship to the Aether Family

The Aether family is a series of preprints by one author, each examining one functional dimension of coordinated agentic software engineering. They are listed in Table 1.1. Descriptions follow the preprints; none has been evaluated in a real deployment, and several are themselves provisional.

Dimension	Focus	Question addressed	Record
AetherGit	Storage and versioning	How are agent-generated changes stored, signed, and versioned?	Baladi (2026a)
AetherFlow	Integration and verification	How are changes integrated into a shared trunk and verified?	Baladi (2026b)
AetherProof	Specification and evidence	How are claims specified and evidenced?	Baladi (2026c)
AetherMesh	Distribution and consistency	How is state synchronized without a central coordinator?	Baladi (2026d)
AetherGovern	Authority and policy	Who may do what, and how is authority revoked?	Baladi (2026e)
AetherRuntime	Monitoring and assurance	What are agents doing while they run?	Baladi (2026f)
AetherOrchestrate	Orchestration	How is work decomposed, assigned, and scheduled?	Baladi (2026g)
AetherIdentity (candidate)	Identity and continuity	Who is the agent, and does that persist?	This paper

Table 1.1. The Aether family and the candidate eighth dimension.

Five observations about the existing records motivate the question. They are statements about the records as the author reads them, and each can be checked against the preprints.

Observation 1 *(observation).* Identity is used everywhere and specified nowhere. AetherGit assumes signed changesets with author keys, AetherMesh's admission includes signature checks,

AetherGovern's authority chains name principals, and AetherRuntime and AetherOrchestrate records are signed by agents. None of the seven preprints specifies how an identity is created, rotated, or ended.

Observation 2 *(observation).* Continuity is not addressed. A key rotation, a migration to another node, or a restart appears in no preprint as a first-class event; an agent that changes its key is, by the existing records, a different principal.

Observation 3 *(observation).* AetherRuntime records signed observations about agents, and AetherOrchestrate records capabilities declared by agents, but neither aggregates observations into a reputation or checks that declared capabilities are truthful (Q5 of AetherOrchestrate).

Observation 4 *(observation).* Sybil resistance is deferred. AetherRuntime and AetherOrchestrate both rely on committees of distinct agents and both list Sybil-resistant sampling as an open question without central identity (Q8 and Q6 respectively).

Observation 5 *(observation).* AetherGovern provides authority chains and revocation. Whether these already cover delegation of scoped authority between identities, which would overlap with the candidate's fifth layer, is not settled by the records.

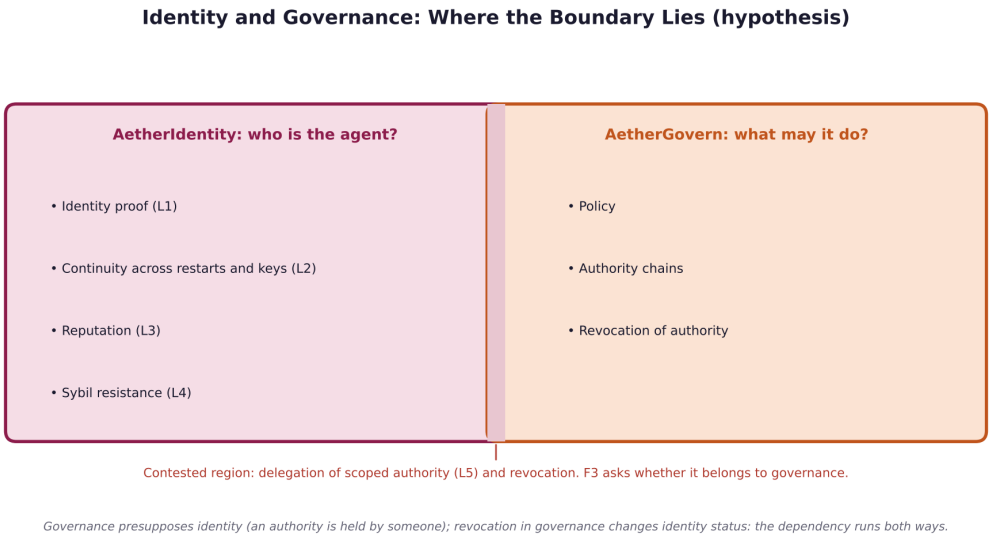


Figure 2 (Fig2_Identity_vs_Governance.png). The conceptual boundary between identity and governance (hypothesis). The draft figure drew a one-way handoff; the redraw shows that the dependency runs both ways and marks delegation and revocation as the contested region.

1.3 The Central Question

Central question *(definition).* Is there a set of requirements for identity proof, continuity, reputation, Sybil resistance, and delegation in multi-agent software systems that (i) is not already specified by the seven existing dimensions, (ii) does not reduce to verification or to governance, (iii) can be specified without reintroducing a central authority, and (iv) yields an observable benefit over existing mechanisms?

The four clauses correspond to the failure conditions: (i) to F1, (ii) to F2 and F3, (iii) to F4, and (iv) to F5. A positive answer needs all four. A negative answer on any one is sufficient to reject the candidate as a separate dimension, though it may survive as a feature of another. The paper does not attempt a positive answer to (iv), which requires an experiment.

1.4 Contributions

1. **A falsification-first procedure**: five failure conditions, six requirements, and a decision rule, stated before the candidate is described (Sections 4 and 5).
2. **A formal framework**: four definitions (identity, continuity chain, reputation record, delegation scope) in a vocabulary compatible with the earlier preprints (Section 3).
3. **Two propositions with proof sketches**: P1 (determinism of local identity issuance) and P2 (irreducible trust anchor), neither machine-checked (Section 3.5).
4. **A five-layer architectural hypothesis** (Section 6), with a classification of which identity functions need agreement and which do not (Section 4.5).
5. **A provisional assessment**: F1–F4 by argument with reservations; F5 untested.
6. **An evaluation design**: nine metrics, five baselines, six scenarios, and a protocol, none executed (Section 8).

1.5 Evidentiary Status

Every statement belongs to one of five categories. A **definition** is a stipulation. An **observation** is a statement about a published record, checkable against it. A **hypothesis** is a claim not established. A **proof sketch** is an argument not machine-checked. A **formula output** is a number computed from a stated formula, not from an experiment. There are no experimental results, no implementation, and no deployment. The paper makes no causal claim about the effect of identity mechanisms on outcomes; comparative language refers to designs on paper.

> **Limitations of this section.** The introduction frames the gap as a question and not as a finding. The five observations rest on the author's reading of preprints written by the same author; independent readers might see coverage that the author does not.

2. Background and Related Work

This section surveys the literatures on which the candidate would draw and states what each leaves open. The review is selective, was not conducted under a protocol, and descriptions of third-party work were not independently verified beyond the cited sources.

2.1 Decentralized Identity and PKI

The dominant way to bind a key to a name is the certificate authority: a trusted party signs a statement that a key belongs to a subject, and verifiers trust a set of root authorities (Cooper et al., 2008). This is efficient and well understood, and it places trust in the roots. Pretty Good Privacy introduced a decentralized alternative, the web of trust, in which users sign one another's keys and each verifier decides whom to trust (Zimmermann, 1995). Decentralized trust management systems such as KeyNote (Blaze et al., 1996) and SPKI (Ellison et al., 1999) shifted attention from names to authorizations, binding permissions directly to keys.

More recent work on decentralized identifiers defines identifiers that a subject controls without a central registry. The W3C specification of Decentralized Identifiers (Sporny et al., 2022) defines a URI scheme, a document that describes verification methods, and resolution through a method-specific mechanism; some methods derive the identifier from a key, others anchor it in a ledger. Sovrin (Sovrin Foundation, 2018) proposed a public permissioned ledger for self-sovereign identity. Self-certifying identifiers, derived by hashing a public key, make the binding between identifier and key checkable without any third party; modern signature schemes such as Ed25519 (Bernstein et al., 2012; Josefsson and Liusvaara, 2017) make the keys compact and verification fast. **What remains open:** these systems specify identifiers and

documents well, but they differ in how they handle key change and recovery, and they say little about reputation or Sybil resistance. Whether an identity protocol for agents can be specified over replicated, causally ordered state in the style of the earlier preprints is a question these works do not address.

2.2 Reputation and Trust

Reputation systems aggregate feedback about past behavior into a signal that guides future interactions (Resnick et al., 2000); Jøsang et al. (2007) survey trust and reputation systems for online services. In peer-to-peer settings, EigenTrust (Kamvar et al., 2003) computes global trust values from normalized local trust by an iteration that converges to the principal eigenvector, and blends in a set of pre-trusted peers to resist collusion. PeerTrust (Xiong and Liu, 2004) adds context and credibility factors. Hoffman et al. (2009) catalogue attacks on reputation systems, among them collusion, slander, and whitewashing. Friedman and Resnick (2001) show that when identities are cheap, a participant with a bad record can discard it and re-enter, which limits what any reputation scheme can enforce unless new identities are costly or newcomers are treated with suspicion.

What remains open: reputation schemes are usually specified as centralized or gossip-based computations with floating-point arithmetic. A scheme whose score may influence admission in a replicated system would have to be a deterministic function of the replicated state, bit for bit, which the cited designs do not address. The pre-trusted peers of EigenTrust also play the role of a trust anchor, which links reputation directly to the anchor problem of Section 3.5.

2.3 Sybil Resistance

Douceur (2002) showed that without a logically centralized authority that certifies identities, a single entity can present multiple identities and subvert a system that counts them, except under unrealistic assumptions about the uniformity of resources and the coordination of entities. Practical responses fall into several families. *Trusted attestation* admits identities only on a trusted party's say-so, which is permissioned. *Resource testing and proof of work* make each identity cost something (Nakamoto, 2008). *Social-graph methods* such as SybilGuard (Yu et al., 2006) and SybilLimit (Yu et al., 2008) assume that the honest part of a social network mixes quickly and has few edges to the Sybil region, and bound the number of Sybil identities accepted per attack edge.

What remains open: each family rests on an assumption (a trusted party, a resource, a graph structure) whose fit for agent populations is unknown. In agent ecosystems, one operator may legitimately run many agents, so 'one entity, many identities' is not in itself an attack; what matters is whether the identities pose as independent. This shifts the question from counting identities to counting operators, and the cryptographic evidence for that is weak.

2.4 Delegation and Capability Systems

Capabilities, introduced by Dennis and Van Horn (1966), are unforgeable references that carry the right to perform an operation; possessing one is authorization. Delegating a capability means passing it on, ideally after narrowing it. Macaroons (Birgisson et al., 2014) are bearer credentials with chained caveats: anyone holding one can add a caveat that restricts it, but cannot remove one, so authority can only narrow along the chain, and verification uses chained message-authentication codes keyed from a root secret. Biscuit (Biscuit project documentation) is a token format with public-key signatures, offline attenuation, and a logic-based language for checks. SPKI and KeyNote, mentioned above, delegate authority along certificate chains.

What remains open: these systems specify attenuation and verification of a single token or chain. They do not specify revocation in a replicated, partitionable system where the verifier may not have seen the revocation yet, and they take the identity of the root as given.

2.5 Research Gap

The literatures supply identifier schemes, reputation algorithms, Sybil defenses, and attenuating credentials, each with its own assumptions and none designed together. The gap this paper investigates is narrow: whether identity, continuity, reputation, Sybil resistance, and delegation can be specified over replicated, causally ordered state, in the style of the earlier Aether preprints, so that the parts that need no agreement are separated from the parts that do, and the irreducible anchor is named. It is a gap in *specification and scoping*, not a claim that existing identity mechanisms are inadequate. Whether it justifies a dimension of its own, or extends an existing one, is what the rest of the paper tests.

Limitations of this section. The review is selective and the gap statement is the author's interpretation. Applied identity and access-management systems, hardware attestation, and the legal and organizational literature on identity were not reviewed, and they may close part of the gap in practice (Q19).

3. Formal Framework

This section gives the vocabulary used in the rest of the paper. All items are definitions or proof sketches. The notation follows the earlier Aether preprints: S denotes the set of deltas a node has admitted, ⊔ the join of replicated state, and A6 the AetherMesh assumption that the admission verdict on a delta is a deterministic function of the delta and its causal closure.

3.1 Definition 1 — Identity

Definition 1 (identity) *(definition).* Fix a signature scheme (Sign, Verify) over a key space, a collision-resistant hash H, and a set Λ of labels (bounded strings). An *inception record* is $\iota = (\text{alg}, \text{pk}, \lambda, t_c)$, signed with the secret key matching pk, where alg names the scheme, $\lambda \in \Lambda$ is a semantic label, and t_c is a creation timestamp. The *identifier* is $\text{id}(\iota) = H(\text{tag} \parallel \text{alg} \parallel \text{pk} \parallel \lambda)$, with tag a fixed domain-separation string. An *identity* is an identifier together with its continuity chain (Definition 2).

Three properties follow by inspection. The identifier is *self-certifying*: anyone holding ι can recompute it and check the signature, with no third party. It is a *deterministic function* of the key, algorithm, and label. And the label is a *claim*, not a verified fact: two agents may choose the same label with different keys and obtain different identifiers. The label gives readers a handle (for example, 'build-agent'); it carries no authority, and label squatting is addressed only by treating labels as non-unique (Q1).

3.2 Definition 2 — Continuity Chain

Definition 2 (continuity chain) *(definition).* A *continuity chain* for identifier id is a sequence $C = (r_0, \ldots, r_n)$, where $r_0 = \iota$ and each $r_k = (\text{id}, k, \text{pk}_k, c_k, h_{k-1}, \sigma_k)$. Here $c_k = H(\text{pk}_{k+1})$ is a *pre-commitment* to the next key, $h_{k-1} = H(r_{k-1})$ links to the previous record, and σ_k is a signature by sk_k over the rest of r_k. C is *valid* if r_0 is correctly self-signed, and for each $k \geq 1$: $H(\text{pk}_k) = c_{k-1}$, h_{k-1} is correct, and σ_k verifies under pk_k. The *current key* is pk_n. Two valid records with the same predecessor and different content are a *fork*; an identity with a fork in S has status *contested*.

DOI: https://doi.org/10.5281/zenodo.24102739

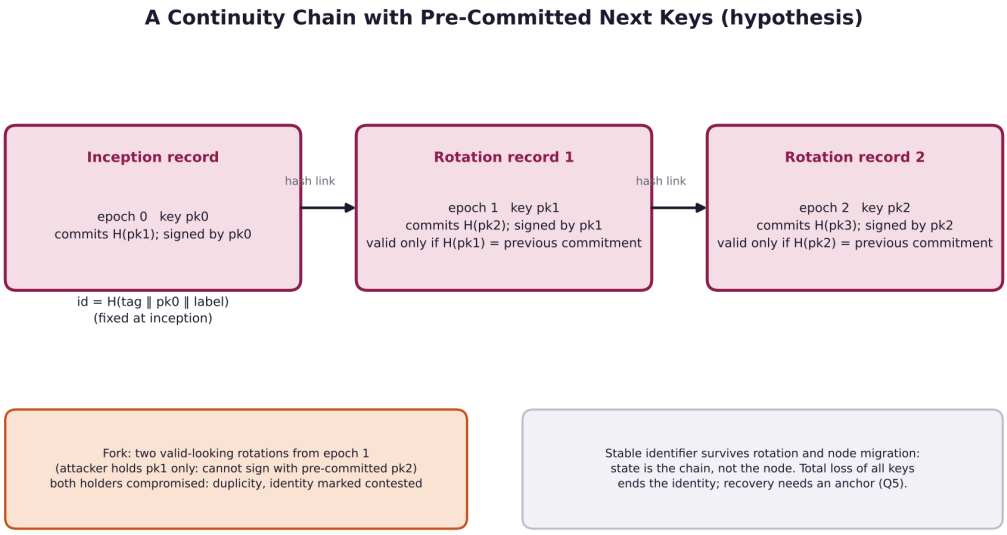


Figure 3 (Fig3_Continuity_Chain.png). A continuity chain with pre-committed next keys (hypothesis). The identifier is fixed at inception and survives rotation and node migration. Fork handling is marked as an open problem.

The pre-commitment is the design choice that matters. An attacker who has stolen only the current secret key can sign as pk_k but cannot produce a valid rotation, because the next record must reveal a key whose hash equals the earlier commitment. Rotation therefore requires the *next* key, which is kept offline. If both are compromised, or if the attacker obtains the next key, there is a fork or a clean takeover, and the protocol can only mark the identity contested and call on an authority outside the chain (Q4, Q5).

Portability. The state of an identity is the chain, which is replicated, not a property of any node. A restart or migration moves the secret key and nothing else, so continuity across nodes (R2) reduces to key custody, and the more subtle problem is preventing two live instances from using the same key at once (Q6).

3.3 Definition 3 — Reputation Record

Definition 3 (reputation record and score) *(definition).* A *reputation record* is $\rho = (o, s, e, v, t)$, signed by the observer identity o, about subject identity s, where e is a content-addressed reference to evidence (for example an AetherRuntime observation, an AetherFlow verification result, or an AetherProof artifact), $v \in \{-1, 0, +1\}$ is the outcome, and t a timestamp. Records are only added to S. Let $n_{os} = \max(\Sigma\, v$ over records from o about s, $0)$ and $C_{os} = n_{os} / \Sigma_j\, n_{oj}$ (with a fixed convention when the denominator is zero). Let p be the uniform distribution over the anchor set Γ_0 (Section 3.5). The *reputation vector* is the fixed point of $t = (1 - a)\cdot C^{T} t + a\cdot p$ for a fixed $a \in (0, 1)$.

This is the structure of EigenTrust (Kamvar et al., 2003), adapted so that the score is a function of the admitted set S. For this to serve in a replicated setting, the computation must be reproducible exactly: the paper therefore specifies a fixed number K of iterations, integer or rational arithmetic, and a canonical ordering of identities, so that all nodes holding the same S obtain the same vector (Appendix B.3). The use of Γ_0 as the damping distribution makes the dependence on the trust anchor explicit. Reputation is a *view* over S; whether it may influence admission is a policy matter (Section 6.3).

DOI: https://doi.org/10.5281/zenodo.24102739

3.4 Definition 4 — Delegation Scope

Definition 4 (delegation scope and chain) *(definition).* Let (Scopes, ⊆) be a family of scopes closed under intersection (a meet-semilattice). A *delegation* is $d = (i, j, S_d, e_d, p_d)$, signed by identity i, granting identity j the scope $S_d \in$ Scopes until expiry e_d, where p_d references the parent delegation (or a root authority record from AetherGovern) under which i itself holds authority. A delegation is *valid* if $S_d \subseteq S_p$, $e_d \leq e_p$, and the parent is valid. The *effective scope* of a chain is the intersection of its scopes and its expiry is the minimum of its expiries. A *revocation* of any link invalidates that link and all links downstream of it.

The definition is an abstraction of attenuation in capability systems (Dennis and Van Horn, 1966; Birgisson et al., 2014): scope and expiry can only narrow along a chain. Expiry is compared with the timestamp carried by the *action* that uses the delegation, not with the verifier's own clock, so that validity does not depend on local time (compare the lease rule of AetherOrchestrate). The residual risk is clock skew between the issuer of the action and the issuer of the delegation (Q16).

3.5 Propositions P1–P2 with Proofs

3.5.1 Determinism of local identity issuance (P1)

The draft statement of P1 is: *if identity issuance is a deterministic function of the public key and the semantic label, the identity layer preserves AetherMesh's determinism requirement (assumption A6).* Three observations.

First, the derivation of the identifier is a hash of its inputs, so it is deterministic for any hash function; the premise holds trivially and carries no information. Second, A6 constrains *admission*, the verdict on a delta, and admission of an identity involves more than computing an identifier: it depends on signatures, on rotation and revocation records in the causal closure, and on whatever evidence (sponsorship, Sybil claims) the protocol uses. Third, the parts of the layer that are naturally local, such as Sybil heuristics based on observed traffic or hardware measurements, are exactly those that would violate A6 if they were consulted at admission. The proposition should therefore say how local evidence is kept out of admission.

Proposition 1 (P1: determinism of local identity issuance, restated) *(stated with proof sketch; not machine-checked).* Suppose (a) $\mathrm{id}(\iota) = H(\mathrm{tag} \,\|\, \mathrm{alg} \,\|\, \mathrm{pk} \,\|\, \lambda)$ with H collision resistant; (b) the admission verdict on every identity-layer record (inception, rotation, delegation, revocation, reputation record, Sybil claim) is a deterministic function of the record and its causal closure, checked by hashes, signatures, and schema only; (c) the derived views, namely identity status, reputation vector, and cluster weights, are deterministic functions of the admitted set S, computed in exact arithmetic with a fixed iteration count and a canonical ordering; and (d) local evidence (traffic observations, hardware measurements, anything not in S) affects replicated state only by causing a node to *create* a signed record. Then (1) admission of identity-layer records satisfies A6, and (2) once all created records have been delivered, every honest node derives the same identifiers, statuses, reputation vector, and cluster weights.

Proof sketch. (1) By (b) the verdict depends only on the record and its closure; local evidence, by (d), is not an argument of the verdict. (2) By AetherMesh's convergence theorem (itself a sketch), honest nodes that have received the same created records hold the same S. By (a) identifiers agree up to hash collisions (Appendix B.1), and by (c) each view is a function of S alone, so the views agree. ∎ (sketch)

What the sketch does not show. (i) Views are not monotone in S (Table 4.2): a status, a score, or a weight may change when more records arrive, so a consumer acting on a view may act on a stale one. (ii) It guarantees determinism, not accuracy: a score or a Sybil weight can be deterministic and wrong. (iii) Recomputing reputation over all of S may be costly (Q11). (iv) Exact arithmetic must be part of the

specification, since floating-point results can differ across platforms. (v) Fork resolution is not solved, only made visible. (vi) The sketch inherits the unverified status of AetherMesh's convergence theorem.

3.5.2 Irreducible trust anchor (P2)

The draft statement of P2 is: *any decentralized identity protocol leaves an irreducible trust anchor: a primitive identity, or set of identities, must be trusted without proof, or the protocol is vulnerable to a bootstrap attack.* We state the argument in the following form.

Admission chain and trust anchor *(definition).* Identity x is *admitted by* identity y if admission of x requires evidence authenticated by y (a signature, an attestation, a sponsorship). A *trust anchor* Γ_0 is a set of identities that a verifier accepts without admission by any other identity.

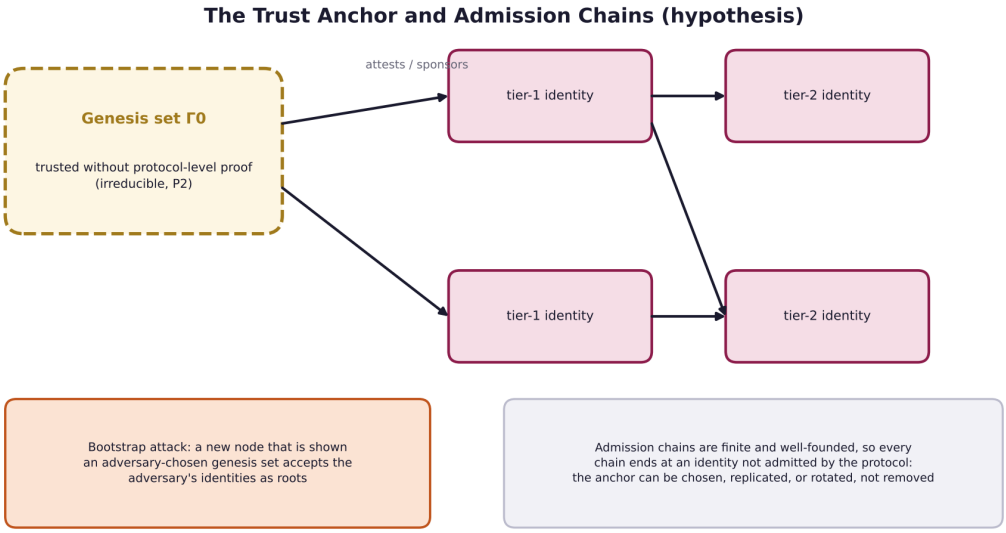


Figure 4 (Fig4_Trust_Anchor.png; supplementary). Admission chains and the trust anchor (hypothesis). Every chain ends in the anchor or in a closed group of identities that vouch for one another; adopting an adversary-chosen anchor is the bootstrap attack.

Proposition 2 (P2: irreducible trust anchor) *(stated with proof sketch; not machine-checked).* In any identity protocol in which an identity is admitted only on evidence authenticated by already admitted identities, the set of identities not admitted by any other identity (or admitted only within a closed group of mutual vouching) is non-empty, and every verifier must accept it by some act that the protocol itself does not verify. If instead admission is unconditional, no evidence is required and Sybil identities are admitted at no cost.

Proof sketch. Consider the 'admitted by' relation among the identities a verifier has admitted. A verifier holds finite state and each admission has finite evidence, so following 'admitted by' backward from any identity either reaches an identity admitted by none, or revisits an identity, in which case the identities on the cycle are admitted only by one another. In either case there is a set that no identity outside it admits. Its members were accepted without evidence from the rest of the system; the protocol cannot verify them without presupposing what it is trying to establish. The verifier must therefore have obtained them out of band (by configuration, by physical or social channels, or by trust in a distributor). If the verifier instead learns the set from the network, an adversary on the path can supply a different set, and its identities become roots; this is the bootstrap attack. ∎ (sketch)

Remarks. (i) The argument is a well-foundedness (regress) argument, not a formal impossibility theorem. (ii) It does not say the anchor must be small, central, or fixed. It can be a threshold set (acceptance requires k of n anchors), it can be rotated by a governed procedure, and its publication can be made

transparent; none of these removes it. (iii) Douceur (2002) establishes a related result for Sybil attacks: without a logically centralized certifying authority, Sybil identities cannot be excluded except under strong assumptions. The two results are different, and both point at the same residue. (iv) The anchor of this proposition is likely to coincide with the root authority of AetherGovern; if so, P2 restates a property of that preprint, which weighs on F3.

Limitations of this section. P1 and P2 remain proof sketches, not machine-checked. P2 is a regress argument and adds no quantitative content. P1 depends on conditions (a)–(d) being respected by every integration, which nothing enforces mechanically (Q19).

4. Failure Conditions F1–F5

This section states the falsification-first procedure and the five failure conditions. Each condition says what would show that the candidate does not deserve to be a separate dimension. Figure 5 summarizes the procedure.

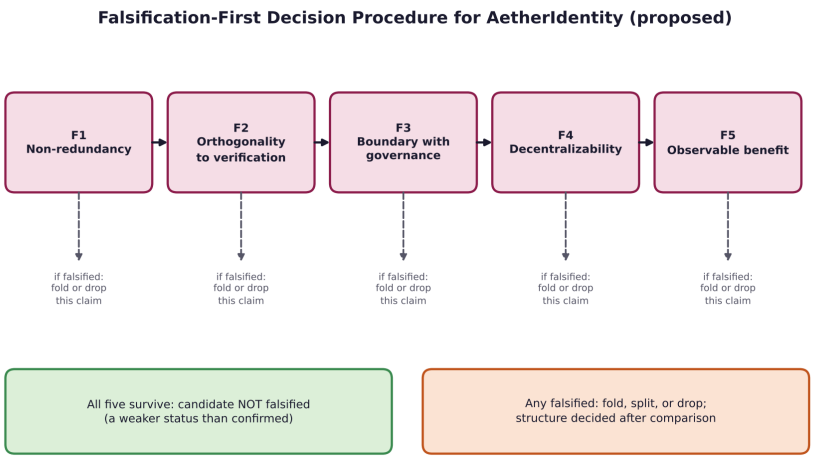


Figure 5 (Fig5_Falsification_Procedure.png; supplementary). Falsification-first decision procedure. Survival of all five conditions yields 'not falsified', which is weaker than 'confirmed'.

4.1 The Problem with Prior Assumptions

A candidate dimension is easy to motivate: any function can be named, given a layer, and shown to be 'missing' from the existing work. The risk is a catalogue of overlapping dimensions, each justified by its own narrative. The remedy is to state, before describing the candidate, the conditions under which it should be dropped or folded, and then to try to meet them. The conditions are the author's own and so are a limited safeguard; their value lies in being explicit and checkable by others.

4.2 The Decision Rule

The rule is fixed in advance. **Not falsified is a weaker status than confirmed.** It means an attempt to falsify the condition, by the stated means, did not succeed; it does not mean the condition holds. The status 'confirmed' is not used for any condition in this paper.

Outcome	Meaning	Consequence for the candidate
Falsified	A condition is met in the sense that it shows redundancy, reduction, centralization, or no benefit	Fold into an existing dimension, narrow the scope, or drop
Not falsified (with reservation)	The attempt did not succeed by the stated means; the reservation says what could change this	Candidate retained for further investigation

Outcome	Meaning	Consequence for the candidate
Untested	The condition needs evidence not available by argument	Candidate cannot be assessed on that condition

4.3 The Five Failure Conditions

F1 — Non-redundancy

Falsified if every requirement R1–R6 is already met by the existing mechanisms or by a bounded extension of exactly one existing dimension. **Attempt.** Each requirement is mapped to the existing preprints (Table 5.1). Signatures and principals exist in several preprints, so R1 is partly covered; R5 and R6 are partly covered by AetherGovern and AetherMesh. R2 (continuity), R3 (reputation), and R4 (Sybil resistance) are addressed by none, and the identity *lifecycle*, the sequence from issuance through rotation to end, is not specified anywhere. **Status: Not falsified (identity lifecycle missing). Reservation:** a bounded extension of AetherGovern that adds identity records, rotation, and revocation of identities might absorb R1, R2, and R5 together; the requirements that would remain (R3, R4) might then be seen as policy inputs to governance and not as a dimension (Q19).

F2 — Orthogonality to verification

Falsified if every identity decision and every trust anchor can be encoded as a verified claim. **Attempt.** Several identity facts are checkable: a signature verifies under a key, a continuity chain is valid, a delegation chain narrows monotonically. These are properties of artifacts and belong with AetherProof. Others are not: the claim that two keys are controlled by one entity is not decidable from cryptographic evidence; a reputation is an aggregate over many observers, not a property of one artifact; and the trust anchor is by P2 not verifiable by the protocol. **Status: Not falsified (narrowly). Reservation:** what is not falsified is the *evidence-based, aggregate, and anchor* part; the artifact-checking half folds into verification. If Sybil claims were required to carry machine-checkable evidence (for example hardware attestation with a verifiable root), part of L4 would fold as well (Q13).

F3 — Boundary with governance

Falsified if identity reduces to governance plus conventional verification. **Attempt.** The two are distinguished by the question each answers (who, versus what may be done, Figure 2), but the boundary is porous. Governance already needs principals, and revocation of authority is naturally revocation of the principal. Delegation of scoped authority (L5) is very close to the authority chains of AetherGovern. **Status: Not falsified (decentralized setting). Reservation:** this is the most fragile condition. If AetherGovern's principals are simply keys and its chains already attenuate, L5 and part of L1 reduce to governance; what would remain distinctive is continuity (L2), reputation (L3), and Sybil handling (L4). P2's anchor is probably also governance's root (Q19, Q18).

F4 — Decentralizability

Falsified if any design that preserves the required properties needs a central authority or strong consistency. **Attempt.** The classification of Section 4.5 shows most identity functions to be monotone or deterministic functions of monotone state, so they need no agreement. Three do not: fixing and changing the trust anchor (P2), recovery after total key loss, and preventing two live instances from using one identity at once. **Status: Not falsified (with residuals). Reservation:** the anchor cannot be removed, only distributed; Sybil resistance beyond what attestation or cost provides rests on assumptions (Douceur, 2002; Yu et al., 2006) whose fit for agents is untested; recovery without an authority is impossible if all keys are lost; and revocation reaches nodes only after a non-zero window. This is the hardest condition.

F5 — Observable benefit

Falsified if no scenario class can be built in which the best existing baseline fails and AetherIdentity succeeds. **Attempt.** None; the condition needs an experiment and the paper reports none. Section 8 proposes baselines, including a composition of existing identity, reputation, and capability mechanisms, and scenarios. **Status: Untested. Reservation:** scenario classes chosen by the proposer risk circularity (Section 9, T5).

4.4 Provisional Assessment

Condition	Status	Reservation (summary)
F1 Non-redundancy	Not falsified (identity lifecycle missing)	R1, R5, R6 partly covered; a bounded extension of AetherGovern might absorb R1, R2, R5
F2 Orthogonality to verification	Not falsified (narrowly)	Artifact checks fold into verification; aggregates, Sybil claims, and the anchor do not
F3 Boundary with governance	Not falsified (decentralized setting)	Most fragile: delegation and revocation overlap; the anchor may be governance's root
F4 Decentralizability	Not falsified (with residuals)	Anchor, Sybil assumptions, total key loss, and cloning cannot be removed; revocation window
F5 Observable benefit	Untested	Requires an experiment; scenario design by the proposer is circular

Table 4.1. Provisional statuses. All are the author's assessment by argument. None is 'confirmed'.

Overall: *not falsified on F1–F4 by argument, each with a reservation, F3 weakest; F5 untested*. This justifies the evaluation design of Section 8; it does not justify a standalone project.

4.5 Monotone and Non-Monotone Identity Functions

Following the CALM perspective (Hellerstein and Alvaro, 2020), identity functions are sorted by whether their output only grows with the input. Table 4.2 is an analysis by the author, not a result.

Function	Monotone in S?	Coordination-free?	Note
Derive identifier	Pure function	Yes	Hash of key, algorithm, label
Issue inception record	Yes	Yes	Self-signed; label is a claim
Verify chain (hashes and signatures)	Pure function	Yes	No local state
Rotate key (append record)	Yes	Yes	Valid only with the pre-committed key
Detect a fork	Yes (sticky flag)	Yes	Two records, one predecessor
Decide which fork branch is genuine	No	**No**	Needs an authority outside the chain
Record a reputation observation	Yes	Yes	Signed by the observer
Compute reputation vector	No (scores move)	Yes, as a function of S	Exact arithmetic required
Record a Sybil claim	Yes	Yes	A claim is evidence-backed, not a proof
Compute cluster weights	No	Yes, as a function of S	Deterministic rule; false positives possible
Issue a delegation	Yes	Yes	Must narrow scope and expiry
Validate a delegation chain	Antitone (revocation shrinks validity)	Yes, as a function of S	View-sensitive within the window
Revoke an identity or link	Record: yes; effect: antitone	Record only	Window until admitted everywhere
Change the trust anchor	No	**No**	Out-of-band or governed (P2)
Recover after total key loss	No	**No**	Needs the anchor

Function	Monotone in S?	Coordination-free?	Note
Enforce one live instance per identity	No	**No**	Consensus-strength (cf. exclusive claims in AetherOrchestrate)

Table 4.2. Classification of identity functions (analysis, hypothesis).

Most identity functions are monotone or deterministic functions of monotone state and need no agreement. Four need something outside the replicated state: choosing the genuine branch of a fork, changing the anchor, recovering from total loss, and enforcing a single live instance. The design aims to confine these four, and the paper records them as the F4 residuals.

Limitations of this section. The conditions are the author's own and were not independently reviewed. The F4 classification is an argument; no protocol was built to test it.

5. Requirements R1–R6

The requirements are the author's statement of what a decentralized identity capability would have to provide. They are definitions, not discoveries, and are stated so that F1 can be applied to them one by one. Each has a testable form that could in principle be checked in an evaluation.

ID	Requirement (definition)	Testable form (planned)	Existing coverage (observation)	Status for F1
R1	Identity proof: an agent can prove it is who it claims to be	A verifier holding the inception record accepts a valid proof of key control and rejects an invalid one; identifier recomputes	Signatures and principals appear in AetherGit, AetherMesh, and AetherGovern; identity records and their lifecycle are not specified	Partly covered
R2	Continuity: identity persists across restarts and transfer between nodes	After rotation, restart, or migration, the identifier is unchanged and the chain verifies (Definition 2)	None of the seven preprints (Observation 2)	Not covered
R3	Reputation: behavioral history is bound to identity	Records about an identifier are retrievable and the score is reproducible from S (Definition 3)	AetherRuntime records observations; no aggregation (Observation 3)	Not covered
R4	Sybil resistance: multiple fabricated identities can be detected	Injected clusters are flagged at a stated error rate, or their combined weight is bounded	Deferred by AetherRuntime and AetherOrchestrate (Observation 4)	Not covered
R5	Delegation: authority can be transferred between identities with bounded scope	Every valid chain narrows scope and expiry; revoked links invalidate downstream links (Definition 4)	AetherGovern authority chains and revocation (Observation 5)	Partly covered (strong overlap)
R6	Partition tolerance: identity functions remain possible when nodes are disconnected	Each partition can verify, rotate, and record on its local S; states converge on healing	AetherMesh converges after partitions; identity under partition is not addressed	Partly covered

Table 5.1. Requirements R1–R6 and their relation to the existing mechanisms (observations and definitions).

R1 (identity proof). Proof of key control is the minimum, and the existing preprints already rely on it. What is not specified is the record that makes an identifier checkable by a verifier that has never met the agent, and the rule for what the label means. **R2 (continuity).** Without a continuity rule, a key rotation is indistinguishable from the appearance of a new agent; the history, reputation, and delegations attached to the old key are lost or have to be re-established by an authority. **R3 (reputation).** The requirement is that the history be *bound* to the identity and *reproducible* from replicated state, not that any particular score be right.

R4 (Sybil resistance). As worded, the requirement asks that fabricated identities can be detected. Douceur (2002) implies that, without trusted certification, detection cannot be guaranteed. The testable form therefore allows either detection at a stated error rate or a bound on the combined influence of a cluster. The requirement is kept as stated and read this way (Reconciliation Notes). **R5 (delegation).** The overlap with AetherGovern is the largest of the six; it is recorded here and bears on F1 and F3. **R6**

(partition tolerance). Each partition must be able to proceed; on healing, conflicting rotations, delegations, or revocations are reconciled by the deterministic rules of Section 6, and anything that cannot be reconciled is marked contested.

Limitations of this section. The six requirements are the author's decomposition of 'identity'. A different list, for example one that adds privacy or unlinkability, would change the F1 mapping; privacy is not addressed in this paper (Q2).

6. Architecture: The Five Layers

This section specifies the hypothesized architecture. All of it is hypothesis, offered so that the failure conditions can be applied to something concrete. Nothing has been implemented and no real-world deployment has been evaluated. The layers answer the five questions of Figure 6. Their order reflects conceptual dependency, not an enforcement hierarchy; each layer reads the replicated state S and appends records to it.

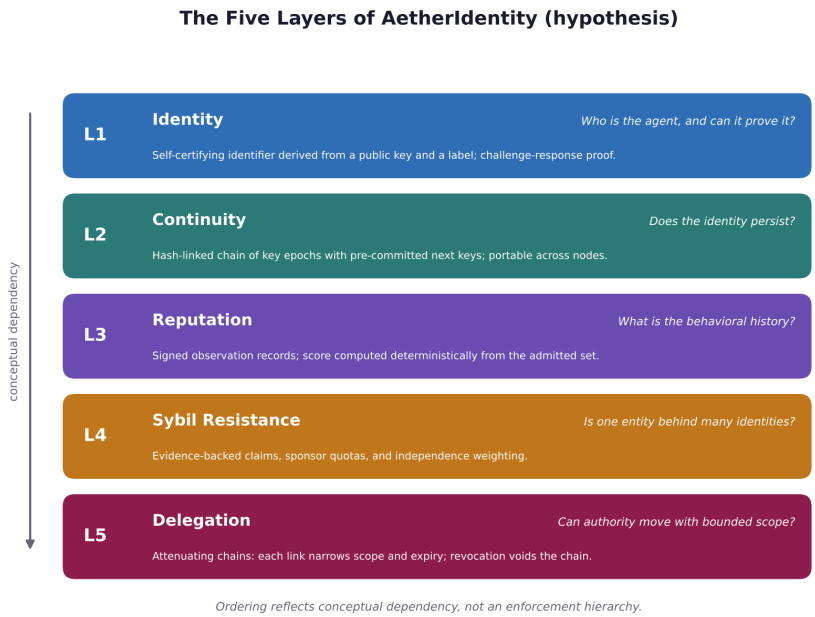


Figure 6 (Fig6_Identity_Layers.png). The five layers of AetherIdentity (hypothesis). Redrawn: the draft's icons were replaced by one-line descriptions of what each layer does.

Layer	Question	Reads from S	Appends to S	Design requirement (hypothesis)
L1 Identity	Who is the agent, and can it prove it?	Inception records	Inception record; sponsor attestation	Self-certifying; label is a claim
L2 Continuity	Does the identity persist?	Chains; revocations	Rotation, recovery, and migration records	Pre-committed next key; forks marked contested
L3 Reputation	What is the behavioral history?	Reputation records; anchor	Reputation records	Evidence-bound; score is a deterministic view
L4 Sybil resistance	Is one entity behind many identities?	Sponsorships; claims	Sybil claims	Claims, not proofs; down-weight, do not exclude
L5 Delegation	Can authority move with bounded scope?	Delegation chains; revocations	Delegation and revocation records	Scope and expiry only narrow

Table 6.1. The five layers. All entries are design hypotheses.

6.1 L1 — Identity

Question. Who is the agent, and can it prove it? **Design.** An agent creates a key pair, forms an inception record (Definition 1), signs it, and publishes it as a delta. Any node can recompute the identifier and verify the signature. Verification of *live* control is by challenge and response: the verifier sends a fresh nonce and the agent returns a signature, or the agent attaches a signature to an action record that is already part of the protocol, which needs no separate exchange.

Existence and standing. Two things are separated. An identifier *exists* if its inception record is admitted; this costs nothing and anyone can do it. It has *standing* if the protocol's admission policy grants it the right to take part (to be counted in committees, to receive tasks, to hold authority). Standing comes from a sponsor, an attestation by an anchored identity, or an AetherGovern decision. The separation is what makes the layer compatible with open systems: existence is permissionless, standing is not, and Sybil resistance (L4) acts on standing.

Algorithm agility. The record names its signature scheme, so that a verifier can reject schemes it no longer trusts, and so that migration to a new scheme can be expressed as a rotation. The paper uses Ed25519 only as an example of a compact scheme (Bernstein et al., 2012). How an identity survives the failure of its scheme is open (Q2). **Hypothetical illustration (not a measurement).** An agent creates key pk_0 with label 'build-agent'. Its identifier is H(tag ‖ alg ‖ pk_0 ‖ 'build-agent'). A second agent with the same label and a different key has a different identifier, and a reader sees two 'build-agent' labels with distinct identifiers; the label carries no authority.

Limitations of this section. The meaning of 'agent' is not defined here: if a hosting platform holds the key, a proof of key control is a proof about the host and not about the agent (Q3). Labels can mislead (Q1). Privacy and unlinkability were not considered.

6.2 L2 — Continuity

Question. Does the identity persist across restarts and key changes? **Design.** The continuity chain of Definition 2 (Figure 3) fixes the identifier at inception and lets keys change by rotation. Events and the records they produce are shown in Table 6.2.

Event	Record produced	What persists	Residual
Restart on the same node	None	Identifier and key	Key custody on the host
Migration to another node	None (key moved) or a rotation (new key on the new node)	Identifier; chain	Two live instances if the old node continues (Q6)
Routine rotation	Rotation record r_k revealing the pre-committed key	Identifier; history; delegations	None beyond clock and delivery
Suspected compromise of current key	Rotation with the offline next key	Identifier; history	Window between compromise and rotation
Compromise of current and next keys	Possible fork	Nothing assured	Contested; needs authority (Q4)
Total loss of all keys	Recovery record (guardians) or none	Only if a recovery set was pre-declared	Impossible without a pre-declared set or the anchor (Q5)

Table 6.2. Continuity events and records (hypotheses).

Recovery. To allow recovery without a central authority, an identity may declare in a rotation record a set of *guardian* identities and a threshold t of n. A recovery record, signed by t guardians, installs a new key after a waiting period during which the current key can veto it. The design trades availability for safety: guardians can collude, the waiting period delays recovery, and the guardians' own identities depend on the anchor. **Upgrades.** A change of code or model weights is not a key event; the identity persists.

Whether it is still 'the same agent' in any useful sense, and whether software measurements should be recorded as evidence, is open (Q7).

Limitations of this section. The probability and consequences of key compromise were not modelled. The guardian design is a sketch, and guardian collusion was not analysed. Cloning (two live holders of one key) cannot be excluded by replicated state alone (Table 4.2).

6.3 L3 — Reputation

Question. What is the agent's behavioral history? **Design.** Reputation records (Definition 3) are signed observations by one identity about another, each referencing evidence. The score is the fixed point of Definition 3, computed from S in exact arithmetic with K iterations. Records without a resolvable evidence reference carry no weight, which limits slander by anonymous or unverifiable assertion, and the weight of an observer's records is itself derived from the observer's score (the EigenTrust recursion).

Iteration count. The iteration map $t \mapsto (1 - a)C^T t + a \cdot p$ is a contraction in the L1 norm with factor $(1 - a)$, because C^T preserves total mass, so the distance to the fixed point shrinks geometrically (Appendix B.3). For a tolerance ε, $K \geq \ln \varepsilon / \ln(1 - a)$ iterations suffice. Table 6.3 gives formula values.

Damping a	Iterations K for ε = 10■■
0.05	270
0.10	132
0.15	86
0.30	39

Table 6.3. Iterations needed for the contraction bound $(1 - a)^K \leq 10^{-6}$ (formula output, not measurement).

Use of reputation. A score is a view over S. It may be used by other layers as an advisory input (for example, in AetherOrchestrate's eligibility ranking), or as an admission input, in which case A6 requires it to be the deterministic function described in P1. The paper recommends the advisory use by default, because recomputation cost and the instability of scores make them poor admission criteria (Q11).

Attack (Hoffman et al., 2009)	Description	Mitigation (hypothesis)	Residual
Whitewashing	Discard a bad identity and re-enter (Friedman and Resnick, 2001)	Newcomers start at zero and need evidence-backed records; standing is sponsor-limited	Cost of new identities may be low in practice
Collusion	A group rates each other highly	Observer weight derived from own score; anchor-weighted damping	Collusive groups close to the anchor
Slander	False negative records	Evidence reference required; per-observer cap	Evidence may itself be fabricated
Sybil boosting	Many identities rate one	Cluster weights from L4	Undetected clusters
Context mismatch	Good at one task, rated for another	Per-domain score vectors	State growth (Q10)

Table 6.4. Reputation attacks and hypothesized mitigations.

Limitations of this section. The aggregation function and its parameters are one example, not a recommendation (Q8). Cold start and whitewashing remain in tension (Q9). No score was computed on real data.

6.4 L4 — Sybil Resistance

Question. Are multiple identities controlled by one entity? **Design.** The layer starts from the observation of Section 2.3: in agent systems, one operator may legitimately run many agents, so the issue is whether identities pose as independent. An identity may declare an *operator*, and independence weighting counts operators and not identifiers. Three mechanisms are combined, each with an explicit assumption.

1 **Sponsor quotas.** Standing is granted by anchored sponsors, each limited to q new identities per epoch, enforced through the sponsor's own sequence numbers: an admission beyond the quota is invalid by a deterministic check, and a sponsor who forks its sequence is detected by duplicity. If an adversary controls s sponsors, the identities with standing that it can create in E epochs number at most s·q·E.

2 **Evidence-backed claims.** Any identity may publish a signed Sybil claim that identities x and y share evidence of common control (shared sponsor and timing, shared attestation, correlated behavior, shared network origin). A cluster is formed from claims signed by at least τ distinct anchored attesters. Each member's weight in committees and in reputation damping is divided by the cluster size, as a deterministic function of S.

3 **Optional cost.** A deployment may add a cost to standing (a resource test or a stake) in the manner of Nakamoto (2008) or Douceur (2002). The paper does not adopt a cost by default, because it excludes resource-poor honest agents and is a policy choice.

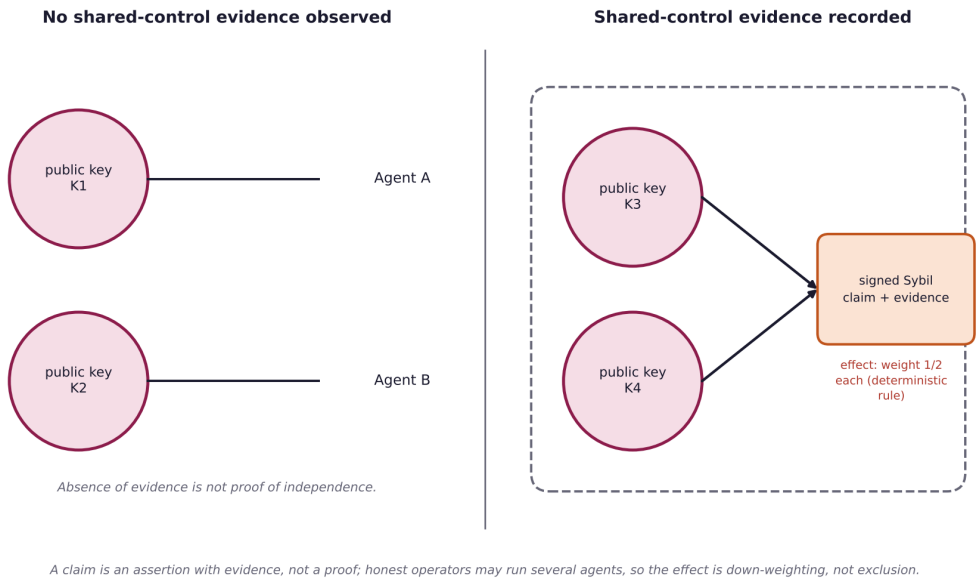


Figure 7 (Fig7_Sybil_Claims.png). Sybil claims (hypothesis). The draft figure marked the left panel 'unique' with a check; the redraw says only that no shared-control evidence was observed, since absence of evidence is not proof of independence. A claim triggers down-weighting by a deterministic rule.

Sponsors s controlled	Quota q	Epochs E	Bound s·q·E
1	10	12	120
5	10	12	600
5	50	12	3000
20	10	52	10400

Table 6.5. Upper bound on identities with standing obtainable by an adversary controlling s sponsors, under the quota rule (formula output; assumes quotas are enforced and sponsors are the only route to standing).

Honest limits. The detection mechanism is a heuristic with false positives: an honest multi-agent operator may be flagged. For this reason the effect of a claim is down-weighting and is reversible by a counter-record, not exclusion. Colluding anchored attesters can fabricate clusters against honest agents. And the quota bound is only as good as the sponsors: it shifts the trust to them and thereby to the anchor (P2).

Limitations of this section. The social-graph methods of Yu et al. (2006, 2008) were not adopted because their assumptions (fast mixing, few attack edges) are untested for agent populations (Q15). The choice of τ, q, and epoch length is open (Q12, Q14), as is the admissible evidence of shared control (Q13).

6.5 L5 — Delegation

Question. Can authority be transferred with bounded scope? **Design.** Delegations (Definition 4) form chains rooted in an authority record from AetherGovern. Each link narrows scope and expiry. An action that relies on a delegation cites its chain; any node can verify the chain from signatures and the records in the action's causal closure, with no shared secret. This differs from Macaroons (Birgisson et al., 2014), whose chained message-authentication codes can be verified only by a party that holds the root secret; public-key signatures make verification available to every node, at the cost of larger tokens and signature checks per link (Q18).

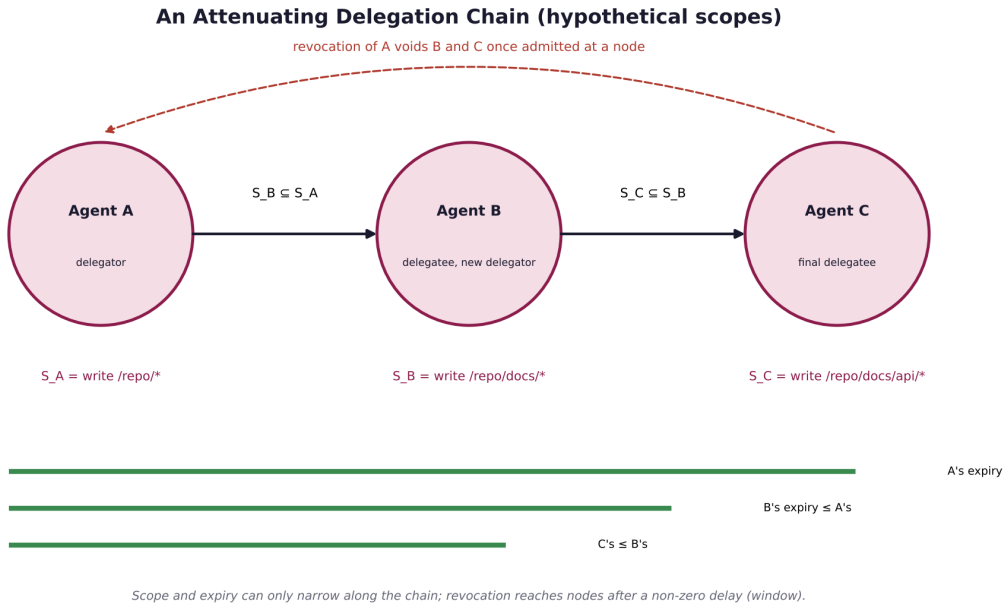


Figure 8 (Fig8_Delegation_Chain.png). An attenuating delegation chain with hypothetical scopes. The draft figure labelled both links with the same scope; the redraw shows scope and expiry narrowing along the chain and revocation reaching nodes after a delay.

Revocation. Revocation records are only added to S. A delegation is invalid for an action if the action's causal closure contains a revocation of that link or of any link above it. This makes the verdict a deterministic function of the action and its closure (A6) and means that an action created before a revocation reached its creator is judged valid, which leaves an exposure window: nodes that have not yet received the revocation continue to honour downstream delegations (compare the window arguments in the earlier preprints). The window is the subject of metric M8. **Depth.** A maximum chain depth bounds verification cost and the blast radius of a compromised link.

Aspect	AetherGovern (author's reading)	AetherIdentity L5 (hypothesis)	Overlap
Root of authority	Authority records and policy	Takes a Govern authority record as the root	Shared root

Aspect	AetherGovern (author's reading)	AetherIdentity L5 (hypothesis)	Overlap
Chain of authority	Causal authority chains	Attenuating delegation chains	**High**
Revocation	Revocation with frontier	Revocation in the closure of an action	**High**
Scope narrowing	Policy-defined	Enforced monotone narrowing	Moderate
Subject of authority	Principals	Identities with continuity and reputation	Low

Table 6.6. Delegation compared with governance. The comparison is the author's reading of AetherGovern and is itself a hypothesis.

Limitations of this section. The scope language is an example. Expressive scopes (as with logic-based checks in Biscuit) make verification harder to bound (Q16). Retroactive versus prospective revocation is a policy choice with consequences not analysed (Q17).

6.6 The Identity Lifecycle

An identity's status is not stored; it is a function of S. Table 6.7 lists statuses and the records that produce them; Figure 9 shows the lifecycle.

Status	Derived when (in S)	Next
Exists	Inception record admitted and verified	Standing
Standing	Sponsored or governed admission with quota respected	Active
Active	Chain valid, no fork, not revoked	Rotated, Contested, Revoked
Rotated (epoch k)	Valid rotation record admitted	Active
Contested	Fork in the chain, or a recovery in its waiting period	Active (resolved) or Revoked
Down-weighted	Member of a Sybil cluster by the deterministic rule	Active (claim countered)
Revoked	Revocation of the identity admitted	Ended
Ended	Revoked, or all keys lost with no recovery set	—

Table 6.7. Identity statuses as functions of the admitted set S (definitions and hypotheses).

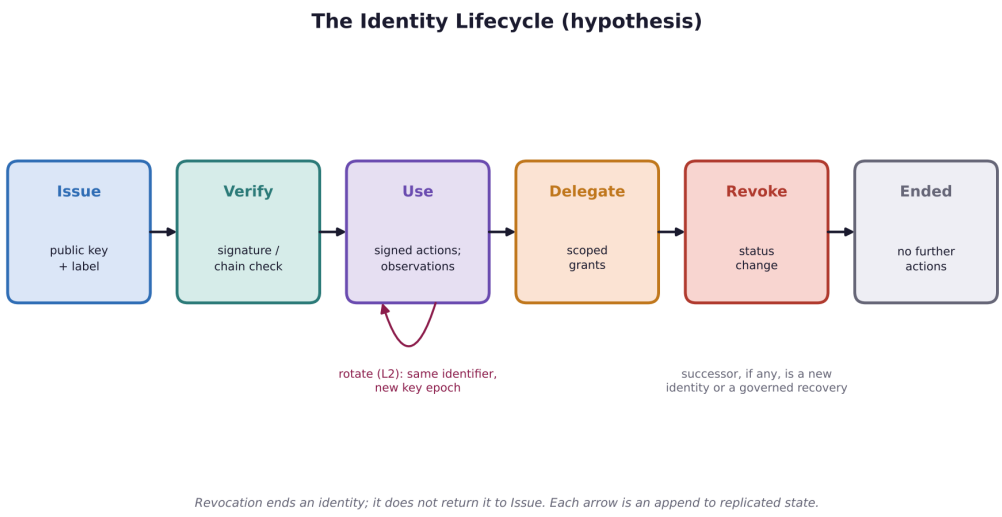


Figure 9 (Fig9_Identity_Lifecycle.png; supplementary). The identity lifecycle (hypothesis). The draft drew a return from Revoke to Issue ('identity ends'), which contradicts revocation; the redraw ends the identity and treats any successor as a new identity or a governed recovery.

Limitations of this section. The lifecycle was not model-checked, and its completeness with respect to real failure modes is not claimed. 'Down-weighted' is a modifier and not a status in the strict sense; it is listed because consumers act on it.

6.7 A Hypothetical End-to-End Trace

To show how the layers compose, Table 6.8 traces one invented agent through its life. The trace is illustrative: it shows which record each step appends and which layer interprets it. It is not a scenario from any deployment, and nothing in it was executed.

Step	Event	Record appended to S	Layer	Derived effect
1	Agent X creates a key pair and publishes its inception record, label 'review-agent'	Inception record	L1	Identifier exists; no standing
2	Sponsor Y, an anchored identity within its quota, attests X	Sponsor attestation	L1, L4	X has standing
3	X completes tasks; AetherFlow verification outcomes are referenced by observers	Reputation records with evidence references	L3	X's score rises from the zero start
4	A governed authority record grants X a broad scope; X delegates a narrower scope to agent Z	Delegation record ($S_Z \subseteq S_X$)	L5	Z may act within S_Z until the earlier expiry
5	X rotates its key on schedule using the pre-committed next key	Rotation record r_1	L2	Same identifier; score and delegation persist
6	Two attesters publish evidence that X and a newer identity W share control	Two Sybil claims ($\tau = 2$)	L4	X and W weights each divided by 2; reversible by counter-record
7	AetherRuntime halts X after an anomaly; governance revokes X's authority	Revocation record	L5, L2	Z's delegation is invalid for actions whose closure contains the revocation; window applies
8	X's operator later wants to continue; its keys are intact	None for the identity; a new governed authority record	L1, L5	Identity may be reinstated by governance, or a new identity issued; the choice is policy

Table 6.8. A hypothetical trace through the five layers (illustration only).

Three things can be read from the trace. First, every step is an append: no layer overwrites anything, so the derivation of status, score, weight, and delegation validity is a function of the accumulated set, which is what makes the determinism argument of P1 plausible. Second, steps 6 and 7 are where agreement-free design is strained: a claim may be wrong, and a revocation reaches nodes with a delay. Third, step 8 shows that reinstatement after revocation is a policy decision, not a protocol property; the identity layer cannot decide whether a revoked agent deserves another chance, which is a governance matter and one more place where the boundary of F3 is not sharp.

7. Interfaces with the Aether Family

This section asks how the candidate would interact with each of the seven existing preprints. Each interface is a hypothesis about what the candidate would consume or produce. None has been implemented and no cross-component interaction has been tested. Descriptions of the existing preprints are the author's reading of the author's own records.

7.1 Interface to AetherGit

AetherGit (Baladi, 2026a) stores signed, content-addressed changes. For the candidate it supplies (i) the signed records in which identity records themselves are stored, and (ii) the history of authorship: a changeset's author key resolves, through the continuity chain, to a stable identifier even after the key has rotated. Without this, an old changeset signed by a retired key would be attributed to a stranger. **Open issue:** verification of historical signatures after a scheme is deprecated (Q2).

7.2 Interface to AetherFlow

AetherFlow (Baladi, 2026b) verifies integrated work. Its verification outcomes are natural evidence for reputation records (a change that passed or failed post-merge verification, with a content-addressed reference). In the other direction, AetherFlow needs to know whose change it is verifying, which the identifier supplies. **Open issue:** a verification outcome says something about a change, not necessarily about the agent that wrote it; mapping outcomes to agents is a modelling choice (Q8).

7.3 Interface to AetherProof

AetherProof (Baladi, 2026c) concerns specification and evidence. Three identity facts are checkable properties and belong there: that a continuity chain is valid, that a delegation chain narrows monotonically, and that a record is correctly signed. Evidence references in reputation records point to AetherProof artifacts. The trusted computing base of an identity operation (the signature library, the hash function) can be listed in the manner of AetherProof's manifests. **Open issue:** the boundary of F2, in particular whether hardware attestation could turn Sybil claims into checkable evidence (Q13).

7.4 Interface to AetherMesh

AetherMesh (Baladi, 2026d) provides convergent propagation and the admission-determinism requirement A6. All identity records are deltas admitted and disseminated by AetherMesh, and every derived view must satisfy the conditions of P1. The candidate needs causal delivery (a rotation cannot be interpreted before its predecessor) and benefits from AetherMesh's partition behaviour (R6). **Open issue:** the dissemination estimates of AetherMesh were validated only partially in that preprint, which limits what can be said about M8 (Q19).

7.5 Interface to AetherGovern

AetherGovern (Baladi, 2026e) provides authority and revocation. The candidate supplies the *subjects* of authority (identities with continuity) and takes a Govern authority record as the root of delegation chains. Revocation of an identity and revocation of authority interact: a revoked identity cannot hold authority, and a revoked authority does not end the identity. **Open issue:** the overlap of delegation and revocation (Table 6.6) and the likely coincidence of the trust anchor with Govern's root; if these were merged, L5 and part of L1 would be absorbed (Q18, Q19).

7.6 Interface to AetherRuntime

AetherRuntime (Baladi, 2026f) records observations about agents and issues halt orders. Its observation records are evidence for reputation records; its halt orders trigger revocation of an identity or of a delegation. Its committees of distinct agents depend on Sybil-resistant sampling, which L4 aims to supply. **Open issue:** reputation derived from Runtime's observations would partly reflect the detectors' errors; and a forged anomaly could be used to harm an honest agent's reputation as well as to halt it (Q11).

7.7 Interface to AetherOrchestrate

AetherOrchestrate (Baladi, 2026g) assigns tasks to agents by eligibility, which depends on capability records and authority. The candidate supplies the identity of the declaring agent, the reputation that can serve as an advisory input to ranking, and the Sybil-resistant sampling that strong-exclusivity committees need (Q6 of that preprint). **Open issue:** misreported capabilities (Q5 of that preprint) are an identity problem only if capability claims are bound to identity and checked against outcomes; the binding is possible, the checking is not specified.

DOI: https://doi.org/10.5281/zenodo.24102739

Component	Identity consumes	Identity produces	Main open issue
AetherGit	Signed append-only store	Stable author identifiers across key changes	Verifying old signatures after scheme deprecation
AetherFlow	Verification outcomes	Subject of each change	Outcome-to-agent mapping
AetherProof	Checkable properties; evidence	Chains and records to check	Boundary of F2
AetherMesh	Convergent propagation; A6	Identity-layer deltas and views	Partial validation of dissemination estimates
AetherGovern	Authority records; revocation	Subjects of authority; chains	Overlap and shared anchor (F3)
AetherRuntime	Observations; halt orders	Reputation; Sybil-resistant sampling	Detector errors in reputation
AetherOrchestrate	Capability records	Identity of declarer; advisory reputation	Truthfulness of capability claims

Table 7.1. Interface summary. All entries are hypothesized interfaces.

Limitations of this section. The interfaces were derived from published descriptions by their own author, not from running code. That seven interfaces each carry an open issue may be read as evidence that the candidate is deeply entangled with the others, which bears on whether it is a separate dimension.

8. Evaluation Design (Proposed)

No evaluation is reported. This section specifies what an evaluation would measure, against which baselines, in which scenarios, and by what protocol, so that the candidate can be retained or dropped on evidence. It is a plan, not a result. The planned experiments compare designs under controlled scenarios and are intended to support statements about differences between designs in those scenarios, not statements about causes in any real deployment.

8.1 Metrics M1–M9

ID	Metric	Definition (planned)	Better is	Relates to
M1	Identity issuance accuracy	Fraction of issued inception records that verify and whose recomputed identifier matches	Higher	R1, P1
M2	Continuity preservation	Fraction of rotations, restarts, and migrations after which the identifier and its history resolve unchanged	Higher	R2
M3	Reputation accuracy	Agreement (for example rank correlation) between computed scores and ground-truth behavior of injected agents	Higher	R3
M4	Sybil detection rate	Fraction of injected Sybil clusters flagged, reported together with the false-positive rate on honest multi-agent operators	Higher	R4
M5	Delegation scope correctness	Fraction of actions authorized exactly when inside the effective scope (no over- or under-authorization)	Higher	R5
M6	Identity verification latency	Time to verify an identity or a chain of given length at a node	Lower	R1, R5
M7	Identity storage overhead	Bytes of identity-layer state per identity and per record	Lower	R2, R3
M8	Revocation propagation latency	Time from revocation record creation to its effect at a quantile q of honest nodes	Lower	R5, F4
M9	Convergence preservation	Number of runs in which honest nodes hold different admitted sets or different derived views after delivery	Must be zero	P1, A6

Table 8.1. Planned metrics. M9 is an invariant (pass or fail), not a quantity to trade against others.

Notes. M4 as supplied is a detection rate; a detector that flags every identity would score perfectly, so a false-positive rate on honest multi-agent operators must be reported with it. M9 is 'must be zero': any

non-zero count would falsify the claim that the identity layer preserves convergence and would take precedence over every other metric.

8.2 Baselines B1–B5

ID	Baseline	Purpose
B1	No identity layer: bare signing keys as principals, with the existing five dimensions	Tests F5 in the simplest case: is any identity layer needed?
B2	Centralized identity provider and certificate authority (X.509 style; Cooper et al., 2008)	Upper reference for latency and revocation; tests the cost of decentralization
B3	Web of trust (Zimmermann, 1995)	Tests whether pairwise trust suffices
B4	DID-based registry anchored in a ledger (Sporny et al., 2022; Sovrin Foundation, 2018)	Tests whether strong-consistency anchoring does better (F4)
B5	Composition of existing components: key-derived DIDs, EigenTrust-style reputation, and Macaroon or Biscuit-style delegation	Strongest baseline: tests F1 and F5 together

Table 8.2. Planned baselines. Each must be given a tuning budget comparable to the candidate's.

8.3 Scenarios S1–S6

ID	Scenario	What it stresses	Baselines expected to be strained (hypothesis)
S1	Key compromise followed by rotation	R2; pre-commitment; window	B1, B3
S2	Sybil flood: many fabricated identities seek standing	R4; sponsor quotas; claims	B1, B3
S3	Partition then merge with conflicting rotations or revocations	R6; fork handling	B2, B4
S4	Migration storm: many agents restart or move at once	R2; cloning; load on verification	B2, B4
S5	Whitewashing: misbehaving agents re-enter under new identities	R3, R4	B1, B5
S6	Delegation abuse: over-broad delegation and revoked intermediate delegators	R5; window	B3, B5

Table 8.3. Planned scenarios. The last column records the author's expectation, a hypothesis the experiments are meant to test, not a result.

8.4 Evaluation Protocol

1. **Pre-registration.** Fix metrics, scenarios, baseline configurations, parameter ranges, and the decision rule before any run, and publish them.
2. **Testbed.** A simulated network of controllable size, latency, loss, Byzantine fraction, and churn, running each design on the same recorded workload and seeds.
3. **Equal effort.** Give each baseline a tuning budget equal to the candidate's.
4. **Fault injection.** Inject key compromises, Sybil clusters, partitions, migrations, whitewashing, and delegation abuse at stated times so ground truth is known by construction; include honest multi-agent operators to measure false positives.
5. **Replication.** Repeat across seeds and network sizes; report distributions, effect sizes, and intervals, not single runs.
6. **Ablation.** Remove each layer in turn (for example, disable pre-commitment or quotas) to test whether it is needed.

7 **Independent scenario review.** Have scenarios reviewed, or proposed in part, by someone other than the proposer.

8 **Decision rule.** F5 is falsified if, for every scenario, some baseline is non-inferior to the candidate on the primary metrics (M2, M4, M5, M8) with an effect-size margin fixed in advance, or if M9 is ever non-zero. Otherwise F5 is 'not falsified', never 'confirmed'.

Limitations of this section. The scenarios and the expected strain on baselines are the author's choices. The effect-size margin is unspecified (Q20). No experiment has been run, and the testbed does not exist.

9. Threats to Validity

Seven threats are listed. They apply to the planned evaluation and to the argument-only assessment in this paper.

ID	Threat	Description	Mitigation (planned)
T1	Construct validity	Injected attacks and simulated agents may not resemble real ones; a Sybil model may be tuned to the detector	Report several metrics; independent review of attack models
T2	Internal validity	Differences may reflect implementation effort or tuning rather than design	Equal tuning budgets; ablations; shared workloads and seeds
T3	External validity	Simulated results may not transfer; no real-world deployment has been evaluated	State scope; replicate on diverse workloads
T4	Baseline strength	A weak or mis-configured baseline would flatter the candidate, notably B5	Use documented configurations; external configuration review
T5	Scenario selection	Scenarios chosen by the proposer may favour the candidate (circularity of F5)	Pre-register; independent scenario proposals
T6	Author bias	The same author proposes the candidate, the conditions, and the assessment	Independent review and replication
T7	Model assumptions	Collision resistance, honest-majority anchors, bounded clock skew, and independence of attesters may fail	Sensitivity runs; state assumptions with every result

Table 9.1. Threats to validity.

Two further limits are specific to this paper. First, F1–F4 are assessed by argument, which can omit a counterexample an implementation would find. Second, as the eighth in a series by one author, its observations about the other seven are not independent evidence: they are the author's consistent reading of the author's earlier work.

10. Open Questions Q1–Q20

The questions are those the author could identify; the list is neither exhaustive nor ordered by importance. They are grouped as in the specification. For each, the text states why it matters.

Identity proof (Q1–Q3)

Q1. *Labels.* The label is a non-unique claim. Should labels be constrained, namespaced, or verified against evidence, and how is squatting handled without a registry?

Q2. *Algorithm agility and privacy.* How does an identity survive the deprecation or break of its signature scheme, including verification of old signatures, and can identifiers be made unlinkable without losing continuity?

Q3. *Agent versus host.* When a platform holds the key, proof of key control is a proof about the platform. What does it take for identity to attach to an agent, as distinct from its operator or host?

Continuity (Q4–Q7)

Q4. *Forks.* When both the current and the pre-committed keys are compromised, the chain forks. What resolution rule, short of an external authority, is acceptable?

Q5. *Recovery.* Is guardian-based recovery safe enough, how are guardians chosen, and what waiting period balances safety and availability?

Q6. *Cloning.* Can two live holders of one key be detected or prevented without consensus-strength coordination (Table 4.2)?

Q7. *Agent upgrades.* When code or model weights change, in what sense is the identity continuous, and should software measurements be recorded as evidence of continuity?

Reputation (Q8–Q11)

Q8. *Aggregation.* Which aggregation function and parameters (damping, anchor weights, observer weighting) are appropriate, and how are verification outcomes mapped to agents?

Q9. *Cold start and whitewashing.* How can newcomers earn standing without making whitewashing cheap (Friedman and Resnick, 2001)?

Q10. *Context.* Should reputation be per task class, and how large does the state become?

Q11. *Stability and cost.* Scores move as records arrive; recomputation is costly. Can reputation be made stable and cheap enough to be used in admission, or should it remain advisory?

Sybil resistance (Q12–Q15)

Q12. *Independence weighting.* How should independence weights be defined, and how is the operator declaration verified?

Q13. *Admissible evidence.* What evidence of shared control should be admissible, can any of it be made machine-checkable (for example through hardware attestation), and how are false accusations bounded?

Q14. *Sponsors.* How should sponsor quotas, epochs, and the threshold τ be set, and how is sponsor capture avoided?

Q15. *Social-graph assumptions.* Do agent populations satisfy the fast-mixing and few-attack-edges assumptions of SybilGuard and SybilLimit (Yu et al., 2006; 2008)?

Delegation (Q16–Q18)

Q16. *Scope language and clocks.* How expressive can scopes be while keeping verification bounded, and what clock-skew bounds does expiry checking need?

Q17. *Revocation semantics.* Should revocation be retroactive or prospective, and how long is the exposure window acceptable?

Q18. *Relation to governance chains.* Are delegation chains and authority chains in AetherGovern the same mechanism, and if so which should own it?

Interfaces and methodology (Q19–Q20)

Q19. *Absorption and shared substrate.* Can a bounded extension of AetherGovern, AetherMesh, or AetherProof absorb R1–R6 (F1)? Is the trust anchor the same as Govern's root, and should the revocation window, which now appears in four preprints, be stated once? Can P1's conditions be enforced mechanically on new predicates? What do applied identity and access-management systems already provide?

Q20. *F5 design and final structure.* What scenario design and effect-size margin would count as observable benefit, set independently of the proposer? And once the Runtime, Orchestrate, and Identity candidates have been assessed under analogous conditions, how should the structure of the family be decided?

Limitations of this section. The questions reflect what the author could identify and may reflect blind spots. Several (Q4, Q6, Q15) are research problems in their own right.

11. Conclusion

We asked whether a set of requirements for identity proof, continuity, reputation, Sybil resistance, and delegation exists that is not already addressed by the seven existing Aether dimensions, does not reduce to verification or governance, can be specified without a central authority, and yields an observable benefit. We stated five failure conditions and six requirements, specified a hypothesized five-layer architecture, restated two propositions as proof sketches, and assessed the candidate by argument.

What the evidence supports

- The seven existing preprints, as the author reads them, use identity but do not specify its creation, rotation, end, or the reputation and Sybil handling attached to it (Observations 1–4).
- Identifier derivation, chain verification, and the derived views can be made deterministic functions of the admitted set, so that admission determinism (A6) is preserved (P1, a sketch).
- Most identity functions are monotone or deterministic functions of monotone state (Table 4.2); four, namely fork resolution, changing the anchor, recovery after total loss, and single-instance enforcement, cannot be had without something outside the replicated state.
- A trust anchor is unavoidable in any identity protocol that admits on evidence (P2, a sketch).

What the evidence does not support

- That the architecture works, performs, or scales: no implementation, experiment, or measurement is reported, and no real-world deployment has been evaluated.
- That the candidate is necessary: F1 and F3 could be refuted by absorption into AetherGovern, and delegation in particular overlaps strongly with its authority chains.
- That Sybil attacks can be detected: detection is heuristic, and the bound of Table 6.5 holds only if sponsors enforce quotas.
- That Tables 6.3, 6.5, B.1, and B.2 are empirical: they are formula outputs under stated assumptions.

Provisional status

F1, F2, and F3 are not falsified, each with a stated reservation, F3 being the most fragile; F4 is not falsified, with residuals; F5 is untested. **Not falsified is a weaker status than confirmed.** No condition is confirmed. Propositions P1–P2 remain proof sketches, not machine-checked. AetherIdentity is a candidate, not a result.

Future work

In order: (1) independent review of the F1–F4 assessment, especially F3 and the overlap of delegation with AetherGovern; (2) a joint treatment of the revocation and coordination windows shared across four preprints, and of the trust anchor shared with AetherGovern (Q19); (3) a systematic review of applied identity systems; (4) a pre-registered simulation study following Section 8 with baselines B1–B5 and scenarios S1–S6; (5) machine-checking of P1 and P2 if the candidate survives. Stopping after step 1 or 2 is a legitimate outcome.

Whether the Aether family should include an identity dimension, and in what structure, is undetermined.

References

Baladi, S. (2026a). AetherGit. Zenodo. https://doi.org/10.5281/zenodo.23966896

Baladi, S. (2026b). AetherFlow. Zenodo. https://doi.org/10.5281/zenodo.23979129

Baladi, S. (2026c). AetherProof. Zenodo. https://doi.org/10.5281/zenodo.24013124

Baladi, S. (2026d). AetherMesh. Zenodo. https://doi.org/10.5281/zenodo.24038038

Baladi, S. (2026e). AetherGovern: A falsification-first investigation of governance in the Aether family. Zenodo. https://doi.org/10.5281/zenodo.24054524

Baladi, S. (2026f). AetherRuntime: A falsification-first investigation of runtime assurance in the Aether family. Zenodo. https://doi.org/10.5281/zenodo.24078477

Baladi, S. (2026g). AetherOrchestrate: A falsification-first investigation of decentralized orchestration in the Aether family. Zenodo. https://doi.org/10.5281/zenodo.24095399

Bernstein, D. J., Duif, N., Lange, T., Schwabe, P., and Yang, B.-Y. (2012). High-speed high-security signatures. *Journal of Cryptographic Engineering*, 2(2), 77–89.

Birgisson, A., Politz, J. G., Erlingsson, Ú., Taly, A., Vrable, M., and Lentczner, M. (2014). Macaroons: Cookies with contextual caveats for decentralized authorization in the cloud. *Proceedings of the Network and Distributed System Security Symposium (NDSS)*.

Biscuit project (n.d.). Biscuit authorization token: specification and documentation. https://www.biscuitsec.org (version and date not recorded).

Blaze, M., Feigenbaum, J., and Lacy, J. (1996). Decentralized trust management. *Proceedings of the IEEE Symposium on Security and Privacy*, 164–173.

Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and Polk, W. (2008). Internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile. RFC 5280.

Dennis, J. B. and Van Horn, E. C. (1966). Programming semantics for multiprogrammed computations. *Communications of the ACM*, 9(3), 143–155.

Douceur, J. R. (2002). The Sybil attack. *Peer-to-Peer Systems (IPTPS)*, LNCS 2429, Springer, 251–260.

Ellison, C., Frantz, B., Lampson, B., Rivest, R., Thomas, B., and Ylönen, T. (1999). SPKI certificate theory. RFC 2693.

Friedman, E. J. and Resnick, P. (2001). The social cost of cheap pseudonyms. *Journal of Economics and Management Strategy*, 10(2), 173–199.

Hellerstein, J. M. and Alvaro, P. (2020). Keeping CALM: When distributed consistency is easy. *Communications of the ACM*, 63(9), 72–81.

Hoffman, K., Zage, D., and Nita-Rotaru, C. (2009). A survey of attack and defense techniques for reputation systems. *ACM Computing Surveys*, 42(1), Article 1.

Josefsson, S. and Liusvaara, I. (2017). Edwards-curve digital signature algorithm (EdDSA). RFC 8032.

Jøsang, A., Ismail, R., and Boyd, C. (2007). A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2), 618–644.

Kamvar, S. D., Schlosser, M. T., and Garcia-Molina, H. (2003). The EigenTrust algorithm for reputation management in P2P networks. *Proceedings of the 12th International World Wide Web Conference (WWW)*, 640–651.

Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. Unpublished white paper.

Resnick, P., Kuwabara, K., Zeckhauser, R., and Friedman, E. (2000). Reputation systems. *Communications of the ACM*, 43(12), 45–48.

Sovrin Foundation (2018). Sovrin: A protocol and token for self-sovereign identity and decentralized trust. White paper.

Sporny, M., Longley, D., Sabadello, M., Reed, D., Steele, O., and Allen, C. (2022). Decentralized identifiers (DIDs) v1.0. W3C Recommendation.

Xiong, L. and Liu, L. (2004). PeerTrust: Supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Transactions on Knowledge and Data Engineering*, 16(7), 843–857.

Yu, H., Kaminsky, M., Gibbons, P. B., and Flaxman, A. (2006). SybilGuard: Defending against Sybil attacks via social networks. *Proceedings of ACM SIGCOMM*, 267–278.

Yu, H., Gibbons, P. B., Kaminsky, M., and Xiao, F. (2008). SybilLimit: A near-optimal social network defense against Sybil attacks. *Proceedings of the IEEE Symposium on Security and Privacy*, 3–17.

Zimmermann, P. R. (1995). *The Official PGP User's Guide*. MIT Press.

Research Status Statement

Formal Architecture Proposal — no implementation claimed. AetherIdentity is a candidate, not a result. It is an architectural hypothesis under investigation, the provisional eighth dimension of the Aether family. No implementation, test, or experiment is reported. No real-world deployment has been evaluated. F1–F4 are not falsified by argument, each with a stated reservation; F5 is untested. Not falsified is a weaker status than confirmed, and no condition is confirmed. Propositions P1–P2 remain proof sketches, not machine-checked. The paper makes no causal claim. The final project structure is undetermined.

Author. Samir Baladi, Interdisciplinary AI Research, Ronin Institute / Rite of Renaissance. ORCID 0009-0003-8903-0029. gitdeeper@gmail.com. **DOI:** https://doi.org/10.5281/zenodo.24102739. **Record ID:** AETHERIDENTITY-01. Version 0.1.0, November 2026. **License:** MIT (code) and CC-BY-4.0 (documentation).

Appendix A — Notation

Symbol	Meaning
(Sign, Verify), H	Signature scheme; collision-resistant hash
$\iota = (alg, pk, \lambda, t_c)$	Inception record: algorithm, public key, label, creation time (Definition 1)
id(ι)	Identifier H(tag ‖ alg ‖ pk ‖ λ)
$C = (r_0, \ldots, r_n)$	Continuity chain of key-epoch records (Definition 2)
$c_k = H(pk_{k+1})$	Pre-commitment to the next key
ρ = (o, s, e, v, t)	Reputation record: observer, subject, evidence, outcome, time (Definition 3)
C_{os}, a, t, K	Normalized local trust; damping; reputation vector; fixed iteration count
Γ_0	Trust anchor set (Section 3.5.2)
$d = (i, j, S_d, e_d, p_d)$	Delegation: from, to, scope, expiry, parent (Definition 4)
S	Set of deltas admitted by a node (its view)
⊔	Join of replicated state
q (quota), s, E, τ	Sponsor quota; sponsors controlled; epochs; attester threshold for a Sybil cluster
β, m	Sybil fraction among standing identities; committee size
A6	AetherMesh assumption: admission verdict is a deterministic function of a delta and its causal closure
F1–F5, R1–R6	Failure conditions; requirements
M1–M9, B1–B5, S1–S6, Q1–Q20	Metrics; baselines; scenarios; open questions

Appendix B — Derivations

All derivations are sketches under stated assumptions. They are included so that the formula outputs in the main text can be reproduced and their assumptions seen.

B.1 Identifier collisions

If H behaves as a random function with b output bits, the probability that any two of n identifiers coincide is at most $n(n-1)/2 \cdot 2^{-b}$ (the birthday bound). Table B.1 evaluates it. The bound concerns accidental collisions; it says nothing about attacks beyond the random-function model, and it fails if H is weak.

Identifiers n	b = 256	b = 128
10^6	4.3e-66	1.5e-27
10^9	4.3e-60	1.5e-21
10^12	4.3e-54	1.5e-15

Table B.1. Birthday bound $n(n-1)/2 \cdot 2^{-b}$ on any identifier collision (formula output).

B.2 Pre-commitment

A rotation to a key pk′ is valid only if $H(pk') = c_k$. An attacker holding sk_k but not sk_{k+1} must therefore find pk′ with $H(pk') = c_k$ and hold its secret key. If H is a random function, each hash evaluation succeeds with probability at most 2^{-b}, so q evaluations succeed with probability at most $q \cdot 2^{-b}$ (a preimage bound). The attacker cannot instead choose its own pk′ and sign: the commitment is fixed by the current record. The argument assumes preimage resistance and that sk_{k+1} is stored separately from sk_k; if both leak, the argument gives no protection. ∎ (sketch)

B.3 Reputation fixed point

Let $T(t) = (1-a)C^T t + a \cdot p$, where C is row-stochastic (each row sums to 1, or is replaced by p when empty). For any vector u, $\|C^T u\|_1 \le \|u\|_1$, since C^T has non-negative entries and unit column sums. Hence $\|T(t) - T(t')\|_1 \le (1-a)\|t - t'\|_1$: T is a contraction, has a unique fixed point t*, and $\|t_K - t^*\|_1 \le (1-a)^K \|t_0 - t^*\|_1 \le 2(1-a)^K$ for probability vectors. Table 6.3 counts the iterations for the factor $(1-a)^K$ alone; the factor 2 adds $\ln 2 / |\ln(1-a)|$ iterations (about 14 for a = 0.05). **Exact arithmetic.** With integer arithmetic at scale 2^S, each iteration adds a rounding error of at most δ per entry; the error recursion $e_{k+1} \le (1-a)e_k + \delta$ gives a limit of δ/a. Integer operations in a fixed order are reproducible across nodes, which is what A6 needs. ∎ (derivation)

B.4 Committee compromise under a Sybil fraction

Suppose committees of m members are sampled uniformly from identities with standing, and a fraction β of them are controlled by an adversary, independently per member (an approximation). For $\beta < 1/2$, the Chernoff–Hoeffding bound gives $P[\text{adversarial members} \ge m/2] \le \exp(-m \cdot D(1/2 \,\|\, \beta))$, with $D(1/2 \,\|\, \beta) = \tfrac{1}{2}\ln(1/(2\beta)) + \tfrac{1}{2}\ln(1/(2(1-\beta)))$. Table B.2 evaluates it. Independence weighting by operator reduces the *effective* β, since a detected cluster of size m counts as weight 1; an undetected cluster counts as m. The table is the reason Sybil resistance matters to the committees of AetherRuntime and AetherOrchestrate; it assumes independent sampling and is not a security guarantee.

β	D(1/2 ‖ β)	m = 50	m = 100	m = 200
0.10	0.5108	8.1e-12	6.5e-23	4.3e-45
0.20	0.2231	1.4e-05	2.0e-10	4.1e-20
0.33	0.0589	5.3e-02	2.8e-03	7.7e-06

Table B.2. Chernoff–Hoeffding bound on the probability that adversarial members reach half of a committee (formula output, independence assumed).

B.5 Delegation chains

If every link satisfies $S_k \subseteq S_{k-1}$ and $e_k \leq e_{k-1}$, then by induction the effective scope (the intersection) equals the last link's scope and the effective expiry equals the last link's expiry, so the chain is checked link by link. Verification costs d signature checks plus d revocation lookups for depth d; M6 is expected to grow linearly in d. **Revocation window.** A revocation of link k reaches node X only after the record is formed, propagated, queued, and verified; the causal argument of the earlier preprints gives $\Delta t \geq t_{form} + t_{prop}(X) + t_{queue}(X) + t_{verify}(X)$ (Baladi, 2026g, Proposition 2). Within this window, X may honour downstream delegations. ∎ (sketch)

B.6 Single live instance and consensus

An agent that wants to ensure that at most one holder of its key is live would need a mechanism by which two candidate holders decide which proceeds. If two holders each claim to be the sole live instance, deciding between them is a two-process consensus problem; by the argument used for exclusive claims in AetherOrchestrate (Appendix B.6 of Baladi, 2026g), it cannot be solved wait-free from convergent replicated state alone (Herlihy, 1991). Only *detection after the fact* (two signed actions with the same sequence number) is available without agreement. The caveats of that argument (model abstraction, Byzantine behaviour) apply. ∎ (sketch)

Reconciliation Notes

This section records the assumptions made and the decisions taken in writing the paper, including where the supplied specification was changed, so they can be reviewed.

Record identifiers. The paper carries DOI 10.5281/zenodo.24102739 as supplied. The specification's identity block gives the same DOI. The seven existing dimensions are cited with the DOIs supplied, and AetherIdentity is treated as the eighth dimension, as in the supplied Figure 1.

Section numbering and order. The specification numbers the title page, abstract, and sections 3–13 consecutively. The paper numbers the main sections 1–11 and follows the specified order: references, research status statement, Appendix A, Appendix B, then these notes. The specification's 'Interfaces' section is Section 7 and covers all seven existing preprints.

Format. Delivered as a PDF only, as requested.

Length. The specification asks for 10,000–15,000 words and 33 or more pages. The delivered document has about 15273 words (as extracted from the PDF, including tables and captions) and 33 pages.

P1 restated. The draft P1 ('if issuance is a deterministic function of the key and label, A6 is preserved') holds trivially for identifier derivation and does not address admission. It was restated around admission, derived views, and the rule that local evidence enters only as signed records. The result is a proof sketch.

P2 as a regress argument. The draft P2 is kept in substance. It is presented as a well-foundedness argument, with the unconditional-admission case stated, and not as a formal impossibility theorem. Its likely coincidence with the root authority of AetherGovern is stated openly.

F-condition statuses. The statuses given were kept: F1 not falsified (identity lifecycle missing), F2 not falsified (narrowly), F3 not falsified (decentralized setting), F4 not falsified (with residuals), F5 untested. Each is accompanied by a reservation. F3 is identified as the most fragile because delegation and revocation overlap strongly with AetherGovern. These are the author's assessments, not independent ones.

R4 reading. R4 as supplied asks that fabricated identities 'can be detected'. Given Douceur (2002), detection cannot be guaranteed; the testable form allows detection at a stated error rate or a bound on the combined influence of a cluster. The paper also notes that one operator may legitimately run many agents.

Metric M4. The detection rate is reported together with the false-positive rate on honest multi-agent operators, since a detector that flags everything would otherwise score perfectly. M9 is treated as a pass/fail invariant.

Open questions. The twenty questions follow the specified distribution (Q1–Q3 identity proof, Q4–Q7 continuity, Q8–Q11 reputation, Q12–Q15 Sybil resistance, Q16–Q18 delegation, Q19–Q20 interfaces and methodology). The question texts are the author's.

No causal claims. Read as: the paper claims no causal effect of identity mechanisms on outcomes. The evaluation design compares designs under controlled scenarios.

Figures. The six supplied figures were redrawn in the style of the earlier records. Figure 1: duplicated names under and inside each hexagon and the label 'AetherFamily' were removed. Figure 2 (identity versus governance): the draft's single handoff arrow, 'identity enables governance', was replaced by a statement that the dependency runs both ways and by an explicit contested region. Figure 6 (layers): icons replaced by descriptions. Figure 7 (Sybil): the 'unique' check mark was removed because absence of evidence is not proof of independence, and a claim is presented as an assertion with evidence. Figure 8 (delegation): the draft labelled both links with the same scope; the redraw shows narrowing and expiry. Figure 9 (lifecycle): the draft returned from Revoke to Issue ('identity ends'), which is contradictory, and was replaced by an end state; rotation was added. Figures 3 (continuity chain), 4 (trust anchor), and 5 (falsification procedure) are supplementary additions.

Illustrative numbers. The hypothetical scopes in Figure 8, the example of Section 6.1, and the values of Tables 6.3, 6.5, B.1, and B.2 are hypothetical or formula outputs.

References. Twenty-nine references are listed: seven Aether records and twenty-two external works. They were assembled from the author's knowledge of the literature; bibliographic details (pages, years, venues) were not individually verified against publisher records while writing and should be checked before citation. The Biscuit entry has no date or version.

Independence. The author wrote the candidate, the failure conditions, and the assessment. The observations about the seven existing preprints are the author's reading of the author's own work. Independent review is the most useful next step.

DOI: https://doi.org/10.5281/zenodo.24102739