Contents

MESH-BFT: Diversity-Weighted Post-Quantum Byzantine Fault Tolerance for Directed Acyclic Meshes

Author: Nenad Vasic

Abstract. We present MESH-BFT, a Byzantine fault tolerant consensus protocol for Directed Acyclic Meshes (DAMs) that provides three novel contributions: (1) diversity-weighted safety, where correlated witnesses are discounted via a multi-dimensional correlation function, raising the effective Byzantine threshold beyond the classical n/3 bound for correlated adversaries; (2) post-quantum security under a formally defined quantum polynomial-time adversary model, where dual-algorithm signing (ML-DSA-65 + SLH-DSA) ensures consensus safety even if one signature scheme is completely broken; and (3) per-record causal finality in O(log n) wall-clock time, independent of total network records, enabling causally concurrent records to finalize in parallel. We prove safety, post-quantum security, and liveness properties, and validate the protocol on a 5-node geo-distributed testnet (March–April 2026, since retired) with measured gossip latency p50=250ms. To our knowledge, MESH-BFT is the first consensus protocol to formally combine diversity-weighted attestation, post-quantum cryptographic guarantees, and per-record finality in a single framework.

---

1. Introduction

Byzantine fault tolerant (BFT) consensus is fundamental to distributed systems, from state machine replication [1] to blockchain protocols [2, 3]. Classical BFT protocols (PBFT [1], HotStuff [2]) assume independent node failures and impose total ordering on all operations, creating leader bottlenecks and O(n) or O(n^2) communication per decision.

DAG-based protocols (DAG-Rider [4], Narwhal/Tusk [5], Aleph [6]) improve throughput by building a directed acyclic graph of proposals and deriving total order from the graph structure. However, they still assume independent Byzantine failures and require global total ordering.

Three critical gaps remain unaddressed in the literature:

Gap 1: Correlated failures. Real-world validators are not independent. Cloud outages, software bugs, and organizational control create correlated failure patterns. Ethereum's client diversity research [14] and the GPoS protocol [9] recognize this problem empirically, but no formal BFT safety proof incorporates validator diversity as a parameter.

Gap 2: Post-quantum security. NIST standardized post-quantum signature schemes (ML-DSA, SLH-DSA) in 2024 [15], but no consensus protocol formally models a quantum adversary

and proves BFT safety under it. DAG-Rider [4] achieves post-quantum safety as a side effect of its hash-based design, without formal quantum adversary analysis.

Gap 3: Per-record finality. Block-based consensus serializes all transactions into blocks, even when they are causally independent. Sui Lutris [7] provides per-object finality for single-writer objects, but shared-state transactions still require full consensus. No protocol proves O(1) finality per record in a general DAG structure.

MESH-BFT addresses all three gaps in a unified framework. Our contributions:

1. **Diversity-weighted safety (Theorem 1).** We define a multi-dimensional correlation function over witness attributes (organization, network topology, geography) and prove that correlated adversaries cannot achieve double settlement — the diversity discount makes it mathematically impossible when k >= 2 correlated witnesses are used.
2. **Post-quantum security (Theorem 2).** We define a quantum polynomial-time (QPT) adversary model for BFT and prove that dual-algorithm signing provides composite security lambda_d + lambda_s under algorithm independence.
3. **Per-record causal finality (Theorem 3).** We prove that records in a DAM achieve confirmation in O(1) attestation rounds and O(log n) wall-clock time, with causally concurrent records finalizing in parallel. Throughput scales linearly with the number of independent witness clusters.

The remainder of this paper is organized as follows. Section 2 defines the system model. Section 3 proves diversity-weighted safety. Section 4 proves post-quantum security. Section 5 proves causal liveness. Section 6 presents empirical evaluation. Section 7 discusses related work. Section 8 concludes.

---

2. System Model

2.1 Directed Acyclic Mesh

Definition 1 (DAM). A Directed Acyclic Mesh is a tuple M = (V, E, Z, tau, sigma) where: - V is a set of records (vertices) - E subset of V x V is a set of directed edges (parent -> child), forming a DAG - Z: V -> {0, 1, ..., N_z - 1} is a zone assignment function over a dynamic zone count N_z (Definition 2) - tau: V -> R+ is a timestamp function - sigma: V -> Sigma* is a signature function mapping records to their cryptographic signatures

Definition 2 (Zone Assignment). For record r with identifier id(r):

```
Z(r) = u64(SHA3-256(id(r))[0..8]) mod N_z
```

where [0..8] denotes the first 8 bytes interpreted as a big-endian unsigned 64-bit integer, and N_z >= 1 is the network's current zone count. N_z is dynamic: it scales with network load under the zone auto-scaler (anchor-attested zone-transition records; hysteresis-damped). Early drafts of this paper fixed N_z = 256 and used only the first hash byte; the shipped runtime (`src/network/zone.rs::for_record_dynamic`) uses the full-entropy 8-byte form above, and retains a 256-zone form solely as a legacy-compatibility path for pre-dynamic-zone records — itself now computed as the full-entropy 8-byte value mod 256 (`for_record`), not from the first hash byte, so the historical first-byte assignment survives only in archived drafts.

Properties: - *Deterministic:* same record always maps to same zone (for a given N_z; zone-count transitions re-route only future records, at an anchor-attested target epoch) - *Uniform:* SHA3-256 output is uniformly distributed, and 2^64 mod N_z bias is negligible for any practical N_z - *Non-manipulable:* id(r) is UUID v7 (timestamp + random); an attacker cannot choose a zone without brute-forcing SHA3

Definition 3 (Causal Dependency). Record r_1 causally precedes r_2 (written r_1 -> r_2) iff: - (r_1, r_2) in E, or - there exists r_3 in V such that r_1 -> r_3 and r_3 -> r_2 (transitive closure)

Records r_1 and r_2 are **causally concurrent** (written r_1 || r_2) iff neither r_1 -> r_2 nor r_2 -> r_1.

Property (Parallel Finality). If r_1 || r_2, their finality processes are independent. No ordering constraint exists between them.

2.2 Identities and Cryptographic Profiles

Definition 4 (Identity). An identity is a tuple I = (pk_d, pk_s, sk_d, sk_s, PoW, epsilon, pi) where: - pk_d, sk_d in {0,1}* are ML-DSA-65 (Dilithium3) public/secret keys - pk_s, sk_s in {0,1}* are SLH-DSA-SHA2-192f (SPHINCS+) public/secret keys (Profile A only) - PoW = (nonce, difficulty) such that leading_zeros(SHA3-256(pk_d || nonce)) >= difficulty - epsilon in {Human, AI, Device, Organization, Composite} is entity type - pi in {A, B, C} is cryptographic profile

Definition 5 (Identity Hash). The identity hash is:

`H(I) = hex(SHA3-256(pk_d))`

Definition 6 (Cryptographic Profiles). - **Profile A (dual-signature):** Signs with both ML-DSA-65 AND SLH-DSA. Verification requires both signatures valid. - **Profile B (single-signature):** Signs with ML-DSA-65 only. Used by browser/constrained nodes. - **Profile C (delegated):** Uses a parent identity's key via delegation record.

Definition 7 (Signature Verification).

`Verify_A(r, I) = Verify_DIL(sigma_d(r), signable(r), pk_d) AND Verify_SPH(sigma_s(r), signab`
`Verify_B(r, I) = Verify_DIL(sigma_d(r), signable(r), pk_d)`

where signable(r) is the canonical deterministic byte representation of r.

2.3 Witness Sets and Attestation

Definition 8 (Witness Set). For record r, the witness set W(r) is the set of identities that have signed attestation records for r:

`W(r) = { I : there exists attestation a such that a.record_id = id(r) AND Verify(a, I) = true }`

Definition 9 (Stake Weight). For identity I with staked balance s(I):

`w(I) = s(I) / S_total`

where S_total = Sum_{I in I_all} s(I) is the total staked supply across all identities.

Definition 10 (Attesting Stake). For record r:

`AS(r) = Sum_{I in W(r)} s(I)`

2.4 Diversity Function

Definition 11 (Witness Profile). A witness profile is a tuple P(I) = (org, subnet, geo) where: - org in Strings — organizational affiliation - subnet in Strings — IP subnet prefix - geo in Strings — geographic zone

Definition 12 (Pairwise Correlation). For witnesses I_i, I_j with profiles P_i, P_j:

`corr(I_i, I_j) = alpha * 1[org_i = org_j] + beta * 1[subnet_i = subnet_j] + gamma * 1[geo_i = geo`

where alpha = 0.5, beta = 0.3, gamma = 0.1 are correlation weights, and 1[.] is the indicator function.

Implementation note (constants, 2026-07-02). The worked examples and tables in this paper use the original specification weights (gamma = 0.1, so the maximum pairwise correlation is 0.9). The shipped runtime (`src/network/consensus.rs`) uses **gamma = 0.2** (maximum pairwise correlation therefore up to 1.0) together with a geographic-bucket ramp that scales gamma up smoothly as a zone gains geographic diversity, rather than a single fixed value. This changes the *numbers* in the tables below but not the *direction of the result*: a larger correlation weight produces a larger diversity discount, so a correlated attacker's effective stake shrinks at least as fast as the tables show. Every safety bound derived here holds under the shipped constants a fortiori. The shipped-constant columns are included alongside the spec-constant numbers in the Section 3.4 table, and the Theorem 1 remark after Step 4 restates the safety bound under corr = 1.0; read the spec-constant numbers as the conservative (weaker-discount) case.

Properties: - corr(I_i, I_j) in [0, 0.9] under the paper's spec constants (up to 1.0 under the shipped gamma = 0.2; maximum when all three attributes match) - corr(I_i, I_j) = α + β = 0.8 when either profile is unknown (conservative: an unregistered witness is treated as maximally correlated short of shared geography — assumed same org and same subnet. This penalizes non-disclosure and prevents a Sybil from gaining independence credit by withholding its profile; treating unknowns as *independent* would let an unprofiled Sybil claim full independence and defeat the diversity mechanism) - Symmetric: corr(I_i, I_j) = corr(I_j, I_i)

Definition 13 (Independence Factor). For witness I_n relative to witness set W:

```
d(I_n, W) = 1 / (1 + Sum_{I_m in W, m != n} corr(I_n, I_m))
```

Properties: - d(I_n, W) in (0, 1] - d = 1 when I_n has zero correlation with all others (fully independent) - d -> 0 as I_n becomes increasingly correlated with more witnesses

Definition 14 (Trust Score). For record r with witness set W(r):

```
T(r) = 1 - Product_{I_n in W(r)} (1 - w(I_n) * d(I_n, W(r)))
```

Properties: - T(r) in [0, 1) - T = 0 when W(r) = empty set - T approaches 1 as more independent, well-staked witnesses attest - Correlated witnesses contribute diminishing marginal trust

Definition 15 (Effective Stake). The diversity-adjusted attesting stake for record r:

```
ES(r) = Sum_{I_n in W(r)} s(I_n) * d(I_n, W(r))
```

This is the key quantity for the safety proof. ES(r) <= AS(r), with equality only when all witnesses are fully independent.

2.5 Confirmation Levels and Settlement

Definition 16 (Confirmation Levels). For record r:

```
level(r) = Anchored    if level(r) >= Confirmed AND r in epoch_seal
                         AND challenge_window(r) elapsed with no open challenge
         = Confirmed   if |W(r)| >= 3 AND clusters(W(r)) >= 3 AND settled_d(r)
         = Attested    if |W(r)| >= 1
         = Unconfirmed otherwise
```

where settled_d(r) is the diversity-weighted settlement predicate of Definition 19 (over the eligible-stake denominator of Definition 18), and challenge_window is 24 hours in the shipped runtime.

Implementation note (confirmation gate). Earlier drafts modeled the third Confirmed conjunct as a tunable scalar trust threshold, `T(r) >= tau_confirm`. No such

parameter exists in the shipped runtime: the third conjunct is the settlement predicate itself (`src/network/consensus.rs`). Three shipped behaviors refine this definition: (i) *Small-network fallback* — when witness profiles are unknown or fewer than 3 distinct organizations are profiled, the diversity-weighted predicate falls back to raw 2/3 attesting stake (Definition 18) over the same denominator; otherwise diversity weighting is mandatory. (ii) *Cross-zone acceleration (shared-witness bridging)* — when ALL of r's cross-zone parents are already finalized, the cluster requirement relaxes from 3 to 1 for r. The witness-count and stake thresholds are never relaxed; the rationale is that the causal chain has already been independently verified in another zone. (iii) *Anchored challenge window* — inclusion in an epoch seal alone does not yield Anchored status; a 24-hour challenge window must elapse with no open equivocation challenge against the record.

Definition 17 (Cluster Detection). Witnesses are partitioned into clusters via union-find: 1. Merge I_i, I_j if org(I_i) = org(I_j) AND subnet(I_i) = subnet(I_j) 2. Merge I_i, I_j if |tau(att_i) - tau(att_j)| <= Delta_timing (500ms timing correlation) 3. Unknown-profile witnesses start as their own cluster, merged only by timing

```
clusters(W) = number of distinct roots after union-find
```

Definition 18 (Settlement, amended). Record r is settled iff:

```
AS(r) * 3 >= S_elig(r) * 2
```

where the eligible-stake denominator is

```
S_elig(r) = S_D(Z(r)) - s(creator(r))
```

and S_D(z) is the settlement denominator for zone z: the registered per-epoch committee stake for z when a committee is active, otherwise the full zone stake (with long-silent stake aged out under liveness decay, floor-guarded). The creator's stake is excluded because a creator is forbidden from attesting its own record — this is enforced at ingest (self-attestations are rejected at insert and retroactively purged), so creator stake is unreachable for the record's quorum and leaving it in the denominator would make the 2/3 threshold unreachable for records created by high-stake identities. The shipped runtime cites this form as "Definition 18, amended" (`src/network/consensus.rs::is_settled`).

Definition 19 (Diversity-Weighted Settlement). Record r achieves diversity-weighted settlement iff:

```
ES(r) * 3 >= S_elig(r) * 2
```

Using effective stake (diversity-adjusted) instead of raw attesting stake, over the same eligible-stake denominator as Definition 18. This is strictly stronger: correlated witness sets must accumulate more raw stake to meet the threshold. (The shipped runtime evaluates this predicate in deterministic fixed-point arithmetic — Q = 10^9 — rather than floating point, so all nodes reach bit-identical settlement verdicts regardless of attestation arrival order or architecture; `is_settled_diverse`, docs/SETTLEMENT-FLOAT-DETERMINISM.md.)

2.6 Adversary Models

Definition 20 (Classical Byzantine Adversary). An adversary A_c that: - Controls f < n/3 identities (by stake weight) - Can deviate arbitrarily from the protocol - Can delay messages but not prevent eventual delivery (partial synchrony) - Cannot forge signatures under the computational hardness of ML-DSA-65

Definition 21 (Correlated Byzantine Adversary). An adversary A_corr that: - Controls f identities, all sharing the same profile P (same org, subnet, geo) - Raw stake weight: Sum

s(I) for I in controlled set - Effective stake weight: Sum s(I) * d(I, W) where d is diminished by correlation

Key insight: Under diversity-weighted settlement, A_corr with stake f needs MORE raw stake than A_c to achieve the same effective stake, because correlated attestations are discounted.

Definition 22 (Quantum Polynomial-Time Adversary). An adversary A_q that: - Has access to a quantum computer capable of running Shor's algorithm - Can break ONE of {ML-DSA-65, SLH-DSA-SHA2-192f} with non-negligible probability - Cannot break BOTH simultaneously (different hardness assumptions: lattice vs hash-based) - All other capabilities same as A_c

Definition 23 (Sybil Adversary). An adversary A_s that: - Can create unlimited identities, each requiring PoW cost >= 2^difficulty - Each identity must stake >= MIN_WITNESS_STAKE (100 beats — the internal, non-tradeable protocol credit) to participate in consensus - Entity clustering with diminishing returns: earning_rate(N) = log_10(1 + N)

---

3. Diversity-Weighted Safety

3.1 Theorem Statement

Theorem 1 (Diversity-Weighted Safety). Let M be a DAM with identity set I, total staked supply S_total, and diversity function d(.,.) as defined in Definition 13. Let r be a record and r' be a conflicting record (asserting a different value for the same logical slot).

If both r and r' achieve diversity-weighted settlement:

```
ES(r)  >= (2/3) * S_total
ES(r') >= (2/3) * S_total
```

Then at least one honest identity must appear in both W(r) and W(r'), contradicting the assumption that honest identities do not attest conflicting records.

3.2 Proof

Step 1: Partition identities.

Let H subset of I be the set of honest identities and B subset of I be the set of Byzantine identities, with |B| controlled by the adversary. Define:

```
S_H = Sum_{I in H} s(I)     (total honest stake)
S_B = Sum_{I in B} s(I)     (total Byzantine stake)
S_total = S_H + S_B
```

Step 2: Bound effective honest stake.

For honest identity I_h with no correlation to other honest identities (best case for the honest set):

```
d(I_h, W) <= 1
```

Therefore: ES_H(r) <= S_H for any record r.

For honest identities that ARE correlated (e.g., two honest nodes from the same organization):

```
d(I_h, W) < 1
```

Therefore: ES_H(r) < S_H in general. Honest diversity matters too.

Step 3: Bound effective Byzantine stake.

The adversary controls B identities. If all Byzantine identities share the same profile (worst case for diversity — maximum correlation):

```
corr(I_i, I_j) = alpha + beta + gamma = 0.9    for all i,j in B
```

For Byzantine identity I_b in witness set W where all other Byzantine identities also appear:

```
d(I_b, W) = 1 / (1 + Sum_{j != b, j in B intersect W} corr(I_b, I_j))
          = 1 / (1 + 0.9 * (|B intersect W| - 1))
```

For |B intersect W| = k Byzantine witnesses from the same profile:

```
d(I_b, W) = 1 / (1 + 0.9(k-1))
```

The total effective Byzantine stake in W:

```
ES_B(r) = Sum_{I_b in B intersect W(r)} s(I_b) * d(I_b, W(r))
```

If all k Byzantine identities have equal stake s_b = S_B/k:

```
ES_B(r) = k * (S_B/k) * 1/(1 + 0.9(k-1))
        = S_B / (1 + 0.9(k-1))
```

Key result: As k grows (more sybil identities), ES_B shrinks. The adversary's effective stake diminishes as 1/k for large k (correlated sybil).

Step 4: Settlement overlap argument.

For record r to achieve diversity-weighted settlement:

```
ES(r) = ES_H(r) + ES_B(r) >= (2/3) * S_total
```

The honest identities attesting r contribute at most S_H effective stake (less if correlated). The Byzantine identities contribute at most ES_B <= S_B / (1 + 0.9(k-1)).

For BOTH r and r' to achieve settlement:

```
ES_H(r)  + ES_B(r)  >= (2/3) * S_total
ES_H(r') + ES_B(r') >= (2/3) * S_total
```

Since honest identities do not attest both r and r' (by assumption), the honest witnesses in W(r) and W(r') are disjoint sets. Therefore:

```
ES_H(r) + ES_H(r') <= S_H    (honest stake is finite, cannot double-count)
```

Adding the two settlement inequalities:

```
ES_H(r) + ES_H(r') + ES_B(r) + ES_B(r') >= (4/3) * S_total
```

But:

```
ES_H(r) + ES_H(r') <= S_H
ES_B(r) + ES_B(r') <= 2 * ES_B_max = 2 * S_B / (1 + 0.9(k-1))
```

Therefore:

```
S_H + 2 * S_B / (1 + 0.9(k-1)) >= (4/3) * S_total = (4/3)(S_H + S_B)
```

Rearranging:

```
S_H + 2*S_B/(1 + 0.9(k-1)) >= (4/3)*S_H + (4/3)*S_B
2*S_B/(1 + 0.9(k-1)) - (4/3)*S_B >= (4/3)*S_H - S_H
S_B * [2/(1 + 0.9(k-1)) - 4/3] >= (1/3)*S_H
```

For k >= 2 (any sybil attack with correlated identities):

```
2/(1 + 0.9(k-1)) <= 2/(1 + 0.9) = 1.053
```

So:

```
S_B * [1.053 - 1.333] >= (1/3)*S_H
S_B * (-0.280) >= (1/3)*S_H
```

This is a contradiction (left side <= 0, right side > 0). Therefore both r and r' CANNOT achieve diversity-weighted settlement when the adversary uses correlated identities (k >= 2). QED.

> **Remark (shipped constants).** Under the shipped runtime weights (gamma = 0.2, so same-profile correlation reaches alpha + beta + gamma = 1.0), the bound tightens: ES_B <= S_B / (1 + 1.0(k-1)) = S_B / k, and the Step-4 coefficient becomes 2/k <= 1 < 4/3 for all k >= 2, so the contradiction S_B * [2/k - 4/3] >= (1/3) * S_H holds a fortiori (the bracket is more negative than the spec-constant -0.280 for every k >= 2). The safety result is therefore strictly stronger under the shipped constants; the spec-constant derivation above is the conservative case. The shipped gamma additionally ramps with geographic diversity (0 below 2 distinct geo buckets, linear up to full gamma at 12 witnesses — `gamma_effective_scaled`); a smaller effective gamma only lowers corr toward the spec value, staying within the two cases analyzed here.

Step 5: Uncorrelated adversary (k=1).

If the adversary uses a single identity (k=1), d = 1, ES_B = S_B. This reduces to classical BFT:

```
S_H + 2*S_B >= (4/3)*(S_H + S_B)
S_H + 2*S_B >= (4/3)*S_H + (4/3)*S_B
(2/3)*S_B >= (1/3)*S_H
2*S_B >= S_H
S_B >= S_H/2
```

Since S_total = S_H + S_B, this means S_B >= S_total/3. The classical BFT bound: the adversary needs > 1/3 of total stake.

Summary: - **Classical adversary (uncorrelated):** Needs > 1/3 of total stake (same as PBFT) - **Correlated adversary (k >= 2 from same profile):** CANNOT achieve double settlement regardless of stake — diversity discount makes it impossible - **Mixed adversary (some correlated, some independent):** Effective threshold lies between 1/3 and impossible, depending on correlation structure

3.2.1 Slot Mutex: Operationalizing the Honest-Attestation Assumption

Step 4 of the proof relies on the premise that **honest identities do not attest conflicting records**. This is an assumption about operator behavior, but consensus cannot depend on behavior alone — it must be mechanized. The **slot mutex** (deployed in v0.7.6 of the reference implementation; see Protocol Whitepaper §11.12) is the mechanism that makes this assumption hold.

Lemma 3.1 (Slot Mutex Enforcement). Let r and r' be records with the same signed `(creator, nonce)` pair. Then no honest node will attest to both r and r', even under adversarial delivery timing.

Sketch. Every v5+ record carries a Dilithium3/SPHINCS+ signature covering a monotonic nonce drawn from the creator's per-identity counter. Two records with the same `(creator, nonce)` therefore share a **slot key**

```
slot_key(r) = hex(SHA3-256(creator_pk(r))) || ":" || hex16(nonce(r))
```

(the shipped runtime materializes the key as a string — the lowercase-hex account hash, a `":"` separator, and the nonce as 16 zero-padded lowercase hex digits — `record.rs::slot_key`). On first ingest of a record r, an honest node writes slot_key(r) → r.id in its slot index S. On subsequent ingest of any record r' with slot_key(r') = slot_key(r), the node verifies a conflict

proof pi = (r, r') — two valid signatures by the same creator over the same nonce — and, on success, writes slot_key(r) into its conflict index C. The attestation path consults C before signing: any record whose slot is in C is rejected from the attestation queue.

Because signature forgery is negligible under the standard Dilithium3/SPHINCS+ unforgeability assumption, the only way for a valid conflict proof to exist is for the creator itself to be Byzantine. An honest node cannot be coerced into attesting both sides of an equivocation by adversarial delivery ordering — it may temporarily attest whichever record arrives first, but once the second arrives, the slot enters C and settlement is blocked for both (see Protocol Whitepaper §11.12, "Settlement gate"). The settlement gate is a hard floor: no record whose slot is in C can be finalized **regardless of attestation weight**, even if the first-seen record already has >2/3 stake-weighted attestations. This closes the race where a Byzantine creator delivers r to a majority of validators fast enough to clear attestation before r' arrives.

Therefore the set of honest attestors to r and the set of honest attestors to r' are mutually exclusive as required by Step 4, *and* neither record can be finalized once the conflict is detected — a strictly stronger property than the theorem requires. The slot mutex thus turns the "honest attestors are disjoint" premise into a structural invariant enforced by every honest node's local state machine, not a behavioral assumption.

An earlier draft of the slot key included the record's zone partition in the tuple, but this was rejected during conflict-injection testing: routing zones are derived from `record_id`, which differs per record by construction, so zone-keyed slots produced distinct keys for equivocating records and defeated the mutex. The canonical slot key is therefore `(creator, nonce)` only — zone-independent.

3.3 Corollaries

Corollary 1.1 (Diversity Advantage). The diversity-weighted settlement strictly dominates classical settlement: any witness set that achieves diversity-weighted settlement also achieves classical settlement, but not vice versa. The adversary's attack space is strictly reduced.

Corollary 1.2 (Honest Diversity Requirement). For honest nodes to achieve settlement, they too must be sufficiently diverse. If all honest nodes share one profile (one org, one subnet):

```
ES_H = S_H / (1 + 0.9(h-1))    where h = number of honest witnesses
                               (spec constants; S_H / h under the shipped corr = 1.0)
```

This creates an incentive for honest witnesses to diversify — exactly the behavior the protocol wants to encourage. Concentration of honest witnesses reduces the system's ability to finalize records.

3.4 Concrete Examples

The following table shows the effective stake ratio for an attacker running k correlated nodes (all sharing the same org, subnet, and geographic zone — maximum correlation). Both constant conventions are shown: the paper's spec constants (corr = 0.9, d = 1/(1 + 0.9(k-1))) and the shipped runtime constants (corr = 1.0, d = 1/k):

Attacker Nodes (k)	d — spec (corr 0.9)	Eff. stake — spec	d — shipped (corr 1.0)	Eff. stake — shipped
1	1.000	100%	1.000	100%
2	0.526	53%	0.500	50%
5	0.217	22%	0.200	20%
10	0.110	11%	0.100	10%
50	0.022	2.2%	0.020	2.0%

Attacker Nodes (k)	d — spec (corr 0.9)	Eff. stake — spec	d — shipped (corr 1.0)	Eff. stake — shipped
100	0.011	1.1%	0.010	1.0%
1,000	0.0011	0.11%	0.001	0.10%

(The k = 5 and k = 10 spec-constant entries correct arithmetic errors in earlier revisions of this table, which printed 0.238/24% and 0.122/12%; 1/(1 + 0.9·4) = 0.217 and 1/(1 + 0.9·9) = 0.110.)

The safety margin grows with the number of correlated sybil identities — the effective stake of a same-profile fleet decays as 1/k under the shipped constants. An attacker with 100 collocated nodes retains ~1% effective stake — diversity discounting renders correlated sybil attacks economically infeasible.

---

4. Post-Quantum Security

4.1 Threat Model

Let Sigma_d = ML-DSA-65 (Dilithium3) and Sigma_s = SLH-DSA-SHA2-192f (SPHINCS+) be the two signature schemes used in Profile A. Let A_q be a quantum polynomial-time adversary (Definition 22) that can break at most one of {Sigma_d, Sigma_s} with non-negligible probability.

Theorem 2 (Dual-Algorithm Security). MESH-BFT consensus remains safe under A_q. Under the independence assumption (Step 1 below), the composite security level is:

```
lambda_MESH = lambda_d + lambda_s
```

where lambda_d and lambda_s are the security levels of the respective schemes. If the independence assumption is relaxed, the conservative fallback bound is:

```
lambda_MESH >= min(lambda_d, lambda_s)
```

4.2 Dual-Algorithm Security Proof

Step 1: Independence of hardness assumptions.

ML-DSA-65 relies on the hardness of Module-LWE (Learning With Errors over module lattices). SLH-DSA relies on the hardness of hash function preimage/second-preimage resistance (SHA-256 based).

These are independent mathematical problems: - Module-LWE is a lattice problem. Best known quantum attack: BKZ with quantum sieving, achieving at most quadratic speedup over classical (Grover-like). - Hash preimage resistance against quantum: Grover's algorithm provides at most quadratic speedup (2^n -> 2^(n/2)).

Claim: An oracle that breaks ML-DSA-65 provides no advantage in breaking SLH-DSA, and vice versa. This follows from the structural independence of lattice problems and hash preimage problems — no known reduction exists between them.

Step 2: Profile A signature security.

A Profile A record r has two signatures: sigma_d(r) and sigma_s(r). Verification requires BOTH to be valid:

```
Verify_A(r) = Verify_d(sigma_d(r), signable(r), pk_d) AND Verify_s(sigma_s(r), signable(r), p
```

To forge a record, the adversary must produce valid (sigma_d*, sigma_s*) for some chosen message m*. Since A_q can break at most one scheme:

Case 1: A_q breaks Sigma_d. The adversary can forge sigma_d* but cannot forge sigma_s. *Therefore Verify_A fails because Verify_s(sigma_s*, m*, pk_s) = false.

Case 2: A_q breaks Sigma_s. The adversary can forge sigma_s* but cannot forge sigma_d. *Therefore Verify_A fails because Verify_d(sigma_d*, m*, pk_d) = false.

Case 3: A_q breaks neither. Standard unforgeability holds.

In all cases, Profile A records are unforgeable. QED.

Step 3: Profile B degradation.

Profile B uses only Sigma_d (ML-DSA-65). If A_q breaks Sigma_d: - Profile B records become forgeable - Profile A records remain secure (Sigma_s still holds)

This creates a trust boundary: Profile B identities are strictly less secure than Profile A under quantum attack.

Mitigation: The protocol can enforce that settlement requires a minimum fraction of Profile A attestations — e.g., at least 50% of attesting stake must come from Profile A identities.

Step 4: Security level.

For Profile A, the adversary must break BOTH schemes. The probability of success:

```
Pr[forge_A] = Pr[break_d AND break_s]
            = Pr[break_d] * Pr[break_s | break_d]
            = Pr[break_d] * Pr[break_s]     (independence)
```

If both schemes have security level lambda (negligible probability 2^(-lambda)):

```
Pr[forge_A] = 2^(-lambda_d) * 2^(-lambda_s)
            = 2^(-(lambda_d + lambda_s))
```

The composite security level is lambda_d + lambda_s. For ML-DSA-65 (NIST Level 3, lambda_d ~ 192) and SLH-DSA-SHA2-192f (NIST Level 3, lambda_s ~ 192):

```
lambda_MESH-A >= min(lambda_d, lambda_s) = 192 bits (primary claim — holds with no independence assumption)
```

Optimistic bound (only under the full break-independence assumption of Step 1):

```
lambda_MESH-A = lambda_d + lambda_s = 192 + 192 = 384 bits
```

The **192-bit `min()` bound is the primary, headline claim**: MESH-BFT Profile A retains full NIST Level 3 (192-bit) post-quantum security as long as *either* signature algorithm remains unbroken, requiring no assumption about the two schemes' break-events. The 384-bit additive figure holds *only* under the strong independence assumption of Step 1 (the two schemes' break-events uncorrelated); bit-level additivity across distinct signature schemes is not a standard composability guarantee, so it is presented as an optimistic upper reference, not the security level operators should rely on.

4.3 Security Level Analysis

Configuration	Schemes	Composite Security	Conservative Bound
Profile A	ML-DSA-65 + SLH-DSA-SHA2-192f	384 bits	192 bits

Configuration	Schemes	Composite Security	Conservative Bound
Profile B	ML-DSA-65 only	192 bits	0 bits if ML-DSA broken
Profile C	Delegated (inherits parent)	Depends on parent profile	Depends on parent profile

Profile A provides belt-and-suspenders security: even under optimistic assumptions about quantum computing progress, an attacker must independently break two fundamentally different mathematical problems. The conservative 192-bit bound exceeds the 128-bit threshold conventionally considered sufficient for long-term security.

4.4 Profile B Trust Boundary

Under quantum attack on ML-DSA-65: - All Profile B signatures become suspect - Profile A signatures remain secure (SLH-DSA still holds) - Mixed witness sets degrade proportionally to their Profile B fraction

The protocol enforces a trust boundary by allowing settlement policy to require a minimum Profile A quorum. For example, requiring >= 50% of attesting stake from Profile A identities ensures that even if all Profile B attestations are forged, the remaining honest Profile A attestations suffice for safety (since 50% of 2/3 > 1/3, preserving quorum intersection).

Note on mixed adversary bounds. The security analysis above treats quantum adversaries (Section 4.1–4.3) and Byzantine adversaries (Theorem 1) independently. The bound for a *mixed* adversary — one that simultaneously controls correlated Byzantine nodes AND possesses quantum computing capability — is conjectured to compose (diversity-weighted stake thresholds apply with quantum-degraded Profile B contributions discounted), but this composition has not been formally proven. A formal treatment of the mixed adversary model is left to future work.

4.5 Algorithm Migration

Corollary 2.1 (Algorithm Migration). If one algorithm shows weakness (security level drops below threshold lambda_min), the network can: 1. Issue ALGORITHM_MIGRATION record with grace period 2. During grace period: both old and new algorithms accepted 3. After grace period: deprecated algorithm rejected 4. Records signed under the deprecated algorithm remain valid (cannot retroactively unsign)

Safety during migration: as long as >= 2/3 of attesting stake has rotated to new keys, settlement remains safe. The dual-signature architecture means the network never has a single point of cryptographic failure — one algorithm can be deprecated while the other continues to provide security.

---

5. Per-Record Causal Finality

5.1 Finality Depends Only on Witness Set

Theorem 3 (Per-Record Causal Finality). For any record r in a DAM M with at least 3 independent witness clusters whose combined stake >= (2/3) * S_total:

1. r achieves Confirmed status in O(1) attestation rounds
2. The confirmation latency is independent of |V| (total records) and |I| (total identities)
3. Causally concurrent records r_1 || r_2 finalize independently and in parallel

Proof.

Step 1: Finality depends only on witness set.

From Definition 16, Confirmed status requires:

```
|W(r)| >= 3 AND clusters(W(r)) >= 3 AND settled_d(r)
```

These conditions depend on: - |W(r)| — number of witnesses for THIS record - clusters(W(r)) — cluster count for THIS record's witnesses - settled_d(r) — the diversity-weighted settlement predicate, computed from THIS record's witness set and the zone's registered stake denominator (a constant-size aggregate lookup)

None of these depend on |V| (how many records exist in the DAM) or |I| (how many identities exist in the network). They depend only on the local properties of r's witness set, plus one O(1) lookup of the zone's pre-maintained stake aggregate for the settlement denominator — which is a per-zone scalar, not a scan over records or identities.

5.2 Attestation Round Complexity

Step 2: Attestation rounds are constant.

Define one "attestation round" as the time for a record to propagate to a witness and for the witness to submit an attestation back. This is bounded by:

```
t_round = t_propagation + t_verify + t_sign + t_return
```

where: - t_propagation = O(log(n) / log(sqrt(n))) * t_hop = O(log n) network hops with sqrt(n) fan-out - t_verify = O(1) — Dilithium3 verification is constant-time (~4ms) - t_sign = O(1) — Dilithium3 signing is constant-time (~5ms) - t_return = O(log n) network hops back

The NUMBER of attestation rounds to reach Confirmed is O(1) — we need exactly 3 diverse witnesses. But each round takes O(log n) for propagation. Total confirmation latency:

```
t_confirm = 3 * t_round = 3 * O(log n) = O(log n)
```

This is per-record latency. In a block-based system:

```
t_confirm_block = O(n) or O(n^2)  (all validators must process all transactions in the block)
```

Per-record is O(log n) vs per-block O(n) or O(n^2). For large n, this is a significant advantage.

Refined statement: Per-record finality requires O(1) attestation rounds and O(log n) wall-clock time, where each attestation round involves gossip propagation. This is independent of |V| (number of records) and depends only logarithmically on |I| (number of nodes) through gossip propagation.

5.3 Parallel Finality for Concurrent Records

Step 3: Parallel finality for concurrent records.

Let r_1 || r_2 (causally concurrent). Their witness sets W(r_1) and W(r_2) are computed independently: - A witness I can be in both W(r_1) and W(r_2) — witnessing one does not prevent witnessing the other - T(r_1) is computed from W(r_1) only; T(r_2) from W(r_2) only - No ordering constraint between r_1 and r_2

Therefore r_1 and r_2 can finalize simultaneously, in parallel. The system's throughput for concurrent records is:

```
throughput = min(|available_diverse_witnesses|, network_bandwidth) * records_per_round
```

This scales linearly with the number of available independent witnesses, unlike block-based systems where throughput is bounded by block size and block time.

Step 4: Comparison to block-based finality.

In a block-based system (PBFT, HotStuff, Tendermint): - ALL transactions in block B must wait for B to be finalized - Block finality requires communication among ALL validators - Conflicting AND non-conflicting transactions are serialized into the same pipeline

In MESH-BFT: - Each record finalizes independently - Non-conflicting records never compete for consensus bandwidth - The “block” equivalent is the individual record

Metric	Block-based (HotStuff)	MESH-BFT
Finality latency per tx	O(block_time + n * round_trip)	O(3 * log(n) * round_trip)
Throughput	O(block_size / block_time)	O(witnesses * records/round)
Independent tx parallelism	None (serialized in blocks)	Full (finalize in parallel)

5.4 Throughput Scaling

Corollary 3.1 (Throughput Scaling). If the network has W diverse witness clusters, and each cluster can process p attestations per second, the theoretical throughput is:

```
max_throughput = W * p    (attestations per second)
```

Each record needs 3 attestations from 3 clusters. Therefore:

```
record_throughput = W * p / 3    (records confirmed per second)
```

This scales linearly with W (number of independent witness clusters), with no upper bound imposed by block size or block time.

5.5 Graceful Degradation

Corollary 3.2 (Graceful Degradation). If fewer than 3 diverse clusters are available: - Records still achieve Attested status (1 witness, lower trust) - No deadlock — the system continues accepting and propagating records - When more clusters come online, records upgrade from Attested to Confirmed

This is strictly better than block-based systems, which halt entirely when fewer than 2f+1 validators are reachable.

5.6 Bounded-View Liveness with Log-Depth Aggregator Chain

Sections 5.1–5.3 establish that a record’s confirmation latency is O(log n) *once an epoch seal is actually proposed by an honest aggregator*. That premise, however, is itself a liveness claim: an epoch must in fact be proposed and finalized within bounded time, even under Byzantine rank-0 proposers. This section formalizes the bound.

System assumptions (reiterated from Section 2).

- **Partial synchrony:** there exists an unknown global stabilization time GST after which message delays between any two correct identities are bounded by some Δ.
- **Honest-majority stake:** the Byzantine stake fraction within any zone is less than 1/3, and the global honest fraction is strictly greater than 2/3.

- **Fresh beacon:** the aggregator chain's chained beacon `beacon(zone, epoch) = SHA3-256(prev_seal_hash || epoch || zone)` is produced from a seal hash the current proposer cannot pre-image under SHA3 collision resistance.

Let `base_timeout = clamp(2 · p95_zone_rtt, 1 s, 10 min)` be the adaptive per-zone timeout. Let `MAX_VIEW_DEPTH = 7` bound the aggregator chain length.

Theorem 4 (Bounded-View Liveness). *Under the system assumptions above and at any time t ≥ GST, every live zone produces and finalizes a seal for its current epoch within*

```
finality_time(zone, epoch) ≤ Σ_{k=0}^{MAX_VIEW_DEPTH-1} base_timeout · 2^k + Δ_cross
                           = (2^MAX_VIEW_DEPTH − 1) · base_timeout + Δ_cross
                           = 127 · base_timeout + Δ_cross
```

where Δ_cross is the cross-zone round latency in the residual case where every rank in the per-zone chain is Byzantine. The probability that cross-zone escalation is needed is bounded above by

```
Pr[all MAX_VIEW_DEPTH ranks Byzantine] ≤ (1/3)^MAX_VIEW_DEPTH ≈ 5 · 10^-4
```

under uniform Byzantine sampling of the stake-weighted rank distribution.

Proof sketch.

Step 1 — beacon entropy. By the chained-beacon construction and SHA3-256 collision resistance, `beacon(zone, epoch)` is computationally indistinguishable from uniform over the output space to any proposer who has not yet observed the previous seal's hash. Therefore the stake-weighted ordering produced by `score(id) = H(beacon || zone || id) / isqrt(stake)` sorts identities in a way no single proposer can bias at proposal time.

Step 2 — uniform Byzantine sampling. Let the zone's staked set be partitioned into honest stake `S_H` and Byzantine stake `S_B` with `S_B / (S_H + S_B) < 1/3`. The priority score is monotone-increasing in a uniform hash, so for a *fixed identity partition* the probability that the rank-k position is occupied by a Byzantine identity is `(S_B^* / (S_H^* + S_B^*))`, the `isqrt`-damped effective-stake fraction. **Caveat (fragmentation): `isqrt` is concave, so it assigns proportionally more weight to small stakes; an adversary who splits `S_B` across many minimum-stake identities can therefore raise `S_B^*` relative to a monolithic honest staker — concavity does *not* by itself preserve the `< 1/3` effective-stake bound.** The bound holds only in conjunction with the minimum-stake admission floor (`MIN_STAKE = 100 beats`, enforced at record admission; Protocol Whitepaper §11.12), which caps the number of distinct Byzantine identities at `S_B / MIN_STAKE` and thereby bounds the achievable `isqrt`-amplification. Under that floor the damped Byzantine fraction stays below 1/3; a tight fragmentation-resistance bound as a function of `MIN_STAKE` is stated here as an assumption and left to the formal model.

Consequently, for each rank independently,

```
Pr[rank k is Byzantine | previous ranks Byzantine] < 1/3
```

and by chaining,

```
Pr[ranks 0, 1, ..., MAX_VIEW_DEPTH - 1 all Byzantine] < (1/3)^MAX_VIEW_DEPTH = (1/3)^7 ≈ 0.0004
```

This is a pessimistic bound — it treats rank occupancy as independent draws, while in practice the rank draws are negatively correlated under a fixed staker set (a Byzantine identity occupying rank k slightly decreases the probability another Byzantine identity occupies rank k+1). The true probability is therefore somewhat lower than (1/3)^7.

Step 3 — per-rank timeout soundness. By the exponential-backoff schedule

```
unlock(k) = (2^k − 1) · base_timeout,
```

at time t = epoch_start_ts + unlock(k) rank k has had the interval [unlock(k−1), unlock(k)] = [2^(k−1) · base_timeout - base_timeout, 2^k · base_timeout - base_timeout] of exclusive proposal authority (or 0 for k = 0). Under partial synchrony with message delay ≤ Δ and base_timeout ≥ 2 · p95_zone_rtt ≥ 2 · Δ after GST, an honest rank-k proposer can (i) observe epoch_start_ts, (ii) construct a valid epoch seal, (iii) propagate it to > 2/3 zone stake, and (iv) collect attestations for finalization within rank k's exclusive window — because the window doubles each rank while Δ remains fixed.

Step 4 — honest rank ⇒ finalization within bounded time. Suppose rank k is the first honest aggregator, with all ranks 0, 1, ..., k − 1 Byzantine or crashed. By Step 3, once rank k's window opens at epoch_start_ts + unlock(k), an honest proposer seals and finalizes the epoch within rank k's window of length 2^k · base_timeout. Total wall-clock bound for this case is

finality_time ≤ unlock(k) + 2^k · base_timeout = (2^(k+1) − 1) · base_timeout.

Step 5 — summing over ranks. The worst per-zone case occurs when k = MAX_VIEW_DEPTH − 1 = 6 (only the last rank is honest):

finality_time ≤ (2^MAX_VIEW_DEPTH − 1) · base_timeout = 127 · base_timeout.

Step 6 — residual case (all ranks Byzantine). In the measure-(1/3)^7 event that every rank is Byzantine, the zone is **stuck** at time t = epoch_start_ts + 127 · base_timeout and cross-zone escalation activates. A **global quorum seal** emitted by any staker in a different zone (with emitter_zone ≠ stuck_zone) achieves finalization once > 2/3 of stake from non-stuck zones attests. By the system assumption that the global honest fraction strictly exceeds 2/3 and the stuck zone contains at most all Byzantine identities, the non-stuck zones retain > 2/3 honest stake and escalation converges in one additional cross-zone round of duration

Δ_cross ≤ p95_cross_zone_rtt + 2 · Δ

which is a finite function of the mesh's inter-zone connectivity and adds a bounded offset to the finality bound.

Step 7 — combining. Taking the union bound over per-zone and cross-zone cases:

finality_time ≤ (2^MAX_VIEW_DEPTH − 1) · base_timeout + Δ_cross
= 127 · base_timeout + Δ_cross,

which establishes the claim. ∎

Corollary 4.1 (Typical-Case Finality). *In the honest-proposer case (rank 0 is honest), the finality bound collapses to finality_time ≤ base_timeout + 0(p95_zone_rtt) — approximately one adaptive timeout plus one gossip round-trip. The shipped runtime sizes the timeout as base_timeout = max(2 · p95_zone_rtt, 5 s) (src/network/zone_rtt.rs), so with the 5-second floor binding in a tight cluster (p95_zone_rtt ≈ 120 ms), honest-proposer finality is roughly 5–8 s wall-clock. A 1-second base_timeout (giving 1–3 s finality) is an illustrative lower-floor configuration, not the shipped value; lowering the floor is an operational tuning decision. Note the worst-case bound is symbolic — 127 · base_timeout — and instantiates to ≈ 635 s + Δ_cross at the shipped floor, not "127 seconds"; it only materializes under sustained adversarial rank selection, which by Step 2 has probability strictly less than 1/3 per rank and decays geometrically with depth.*

Corollary 4.2 (Independence from Zone Size). *The finality bound 127 · base_timeout + Δ_cross is independent of the number of identities in the zone. This is a direct consequence of bounding view depth at a constant MAX_VIEW_DEPTH = 7 rather than at f + 1: finality latency does not scale with stake concentration or staker count, preserving the O(log n) per-record latency of Theorem 3 even under Byzantine rank-0 proposers.*

Remark on depth selection. MAX_VIEW_DEPTH = 7 yields residual escalation probability (1/3)^7 ≈ 4.57 · 10^-4. Doubling depth to 14 would reduce this to (1/3)^14 ≈ 2 · 10^-7

but would *also* raise the per-zone worst case from 127 · base_timeout to (2^14 − 1) · base_timeout ≈ 16383 · base_timeout — an unfavorable trade for a rare residual that cross-zone escalation already handles in O(Δ_cross). Depth 7 is the knee of the curve: exponentially rare residual *and* bounded worst-case latency.

Remark on liveness slashing. Theorem 4 is a worst-case bound that assumes Byzantine rank-0 is *silent*. An actively hostile rank-0 that proposes malformed seals is trivially rejected at ingest (signature, Merkle-root, or zone-registry verification failure) and the rank advances immediately. Silent rank-0 is priced by the 1 %-stake liveness-failure slash (Protocol Whitepaper §11.12.3 Part D), which turns long-run silence from zero-cost to negatively-rational — no rational operator sustains rank-0 inactivity across multiple epochs.

---

6. Evaluation

> **Historical evaluation (March–April 2026).** Every measurement in this section comes from the project's first testnet, retired when its VPS fleet was decommissioned on 2026-06-09. Provider names and city locations are omitted per publication policy; node labels are generic (VPS-EU-1 … VPS-US-2, Laptop). Quantities are stated in **beats**, the non-tradeable protocol credit (internal code symbol "TYOS" at measurement time; renamed the beat 2026-07), and the "faucet" was that era's participation faucet (credits earned by running a node) — never a public token distribution. These are point-in-time historical measurements, not claims about a currently running network.

6.1 Testnet Deployment

We deployed a 5-node geo-distributed testnet across 3 hosting environments (two commercial VPS providers plus one residential machine) and 2 continents.

Node	Location	Hosting	Role	Latency to genesis
VPS-EU-1	EU (North)	Commercial VPS	Genesis + Witness	—
VPS-EU-2	EU (Central)	Commercial VPS	Witness	~20ms
VPS-US-1	US (West)	Commercial VPS	Witness	~150ms
VPS-US-2	US (East)	Commercial VPS	Witness	~120ms
Laptop	EU (local)	Residential dev machine	Witness (NAT, push-only)	~5ms

Network characteristics: - EU-to-EU: ~20ms RTT - EU-to-US: ~120-150ms RTT - Gossip fan-out: sqrt(5) = 2 peers per hop - Maximum propagation: 2 hops to reach all nodes

At the time of measurement, the testnet had processed 15,381 records across 64 accounts with 2,401 beats staked. Supply conservation was verified at 10B beats (10^19 base units) on all 5 nodes. Nodes maintained 14.9 hours continuous uptime with 3 active peers each.

6.2 Gossip Propagation Latency

Measured from the genesis node (VPS-EU-1) over 50 push measurements:

Percentile	Latency
p50	250ms
p90	500ms
avg	749ms

The average is skewed by 5 cross-Atlantic outliers exceeding 1 second. The EU-to-EU path (~20ms RTT) dominates the fast case; EU-to-US (~150ms) adds the tail latency. With sqrt(n) fan-out, propagation scales as O(log n / log sqrt(n)) hops.

End-to-end record finality timing (from a 60-second test session): - Identity creation (PoW): ~10s desktop, ~30-60s phone - Faucet claim: <1s - First transfer completion: 3-4s - Full flow (incognito browser to send): under 30 seconds

Cross-network operations verified: - Transfer VPS-EU-1 to VPS-US-2: visible on all 5 nodes - Transfer VPS-US-2 to Laptop: visible on all 5 nodes - Transfer phone (Profile B) to laptop: received - Proof VPS-EU-1 to VPS-US-2: verified (`balance_range` — the era log labeled this "Groth16", incorrectly: the shipped construction was and is the Phase-1 SHA3-256 commitment scheme; Groth16 is design-stage and wire-rejected) - Proof VPS-US-2 to VPS-EU-1: verified (`content_commitment`, same Phase-1 construction)

6.3 Projected Scaling

Using the measured sqrt(n) gossip model, we project confirmation latency as the network grows:

Nodes (N)	Hops (log N / log sqrt(N))	p50 Latency	p90 Latency	Confirmation (3 witnesses)
5	2	250ms	500ms	~1s
50	4	500ms	1s	~2s
500	6	750ms	1.5s	~3s
5,000	8	1s	2s	~4s
50,000	10	1.25s	2.5s	~5s

Confirmation latency grows as O(log n), not O(n). At 50,000 nodes, estimated confirmation time is ~5 seconds, compared to block-based systems that scale linearly with validator count.

Throughput scales linearly with the number of independent witness clusters:

Witness Clusters (W)	Attestations/sec/cluster	Records Confirmed/sec
3 (2026-03 testnet)	100	100
10	100	333
50	100	1,667
100	100	3,333
1,000	100	33,333

Each node processes attestations at ~100/second, limited by Dilithium3 verification at ~4ms per signature. Throughput has no block size bottleneck.

6.4 Diversity Discount Impact

To validate the theoretical diversity discount, we compute the effective stake ratio for a correlated attacker with k nodes sharing the same organizational affiliation, network subnet, and geographic zone (maximum pairwise correlation; spec constants, corr = 0.9 — see Section 3.4 for the shipped-constant columns, corr = 1.0):

Attacker Nodes (k)	Independence d(I_b)	Effective Stake Ratio	Discount
1	1.000	100%	0%
2	0.526	53%	47%
5	0.217	22%	78%
10	0.110	11%	89%
50	0.022	2.2%	98%
100	0.011	1.1%	99%
1,000	0.0011	0.11%	99.9%

(The k = 5 and k = 10 rows correct arithmetic errors in earlier revisions, which printed 0.238/24% and 0.122/12%.)

The effective stake ratio decays as approximately 1/k for large k: a correlated sybil fleet gains almost no effective stake from additional nodes. An attacker deploying 100 collocated nodes controls only 1.1% effective stake regardless of node count.

6.5 Zone Distribution

With Z(r) = SHA3-256(id(r))[0] mapping records to 256 zones (the legacy first-byte assignment, in effect during the experiment; Definition 2 now specifies the dynamic full-entropy form) and 15,381 total records: - Expected records per zone: 15,381 / 256 ~ 60 (uniform distribution) - SHA3-256 uniformity guarantees that zone assignment is non-manipulable — an attacker cannot select a target zone without brute-forcing the hash function

6.6 Conservation Invariant

The token supply conservation invariant was verified continuously across all test scenarios: - Normal operation: all 5 nodes report identical supply (10,000,000,000,000,000,000 base units = 10B beats) - Node failure and restart: killed VPS-US-2, created records, restarted — VPS-US-2 caught up (14,644 records), supply consistent - Genesis failure: killed VPS-EU-1 (genesis node) — network continued (VPS-EU-2 created records, VPS-US-1 received). Supply consistent on restart - Cross-network transfers: no violation detected across 15,381 records processed

6.7 Comparison to Prior Systems

System	Throughput (TPS)	Finality Latency	Nodes Tested	Diversity Model	PQ-Safe	Per-Record
PBFT [1]	~1,000	1-5s	4-20	No	No	No
HotStuff [2]	~2,000	3-5s (3 rounds)	4-100	No	No	No
Narwhal/Tusk [5]	130,000	<3s	50	No	No	No

System	Throughput (TPS)	Finality Latency	Nodes Tested	Diversity Model	PQ-Safe	Per-Record
Bullshark [13]	130,000	<2s	50	No	No	No
Avalanche [3]	4,500	1-2s	2,000	No	No	Partial
Sui Lutris [7]	300,000	<500ms	100	No	No	Owned only
MESH-BFT (testnet)	~100*	~1s*	5	**Yes**	**Yes**	**Yes**
MESH-BFT (projected)	~33,000*	~3s*	1,000	**Yes**	**Yes**	**Yes**

*Estimated from measured attestation rates and projected gossip model. Actual benchmarks pending large-scale simulation.

Our raw throughput is lower than Narwhal/Tusk and Sui Lutris because MESH-BFT does not batch records into blocks. The tradeoff is per-record granularity and the elimination of total ordering requirements. For applications that require per-record finality — sensor data, document signing, identity attestation, IoT provenance — this is the appropriate design point. MESH-BFT is the only system in the comparison that provides all three of diversity-weighted consensus, post-quantum security, and per-record finality.

---

7. Related Work

7.1 Classical BFT

PBFT [1]. Castro and Liskov introduced practical BFT with a leader-based, three-phase protocol (Pre-Prepare, Prepare, Commit) achieving O(n^2) communication in the normal case and O(n^3) during view changes. Safety relies on quorum intersection — two sets of 2f+1 replicas must overlap in at least f+1 honest nodes. PBFT is limited to approximately 20-100 nodes due to quadratic messaging, requires a designated leader, and has no model for correlated failures. MESH-BFT eliminates the leader, view changes, and total ordering requirement, while adding diversity discounting that has no PBFT equivalent.

HotStuff [2]. Yin et al. achieved linear BFT (O(n) in both normal and view-change cases) via quorum certificates and a three-chain commit rule. The key innovation is that view changes are as cheap as normal operation. However, HotStuff remains per-block and leader-based, and threshold signatures require trusted setup. MESH-BFT is leaderless with per-record finality and no threshold signature dependency.

Tendermint [16]. Buchman formalized a BFT consensus engine for blockchain with immediate finality (no forks), partial synchrony, and O(n^2) gossip-based communication. Tendermint powers the Cosmos ecosystem but shares the leader bottleneck and per-block finality limitations of classical BFT. MESH-BFT's gossip layer draws inspiration from Tendermint's approach but operates on individual records rather than blocks.

7.2 DAG-Based BFT

DAG-Rider [4]. Keidar et al. proposed a two-layer design where reliable broadcast builds a DAG and local interpretation produces total order with zero extra messages. Safety relies only on hashes, making it post-quantum safe as a side effect — though without a formal quantum adversary model. DAG-Rider achieves O(n) amortized communication per transaction and is optimal in terms of message complexity. MESH-BFT differs in that we do not require total ordering and we formalize the post-quantum property with an explicit QPT adversary model.

Narwhal and Tusk [5]. Danezis et al. separated reliable dissemination (Narwhal mempool) from ordering (Tusk consensus), achieving 170,000 tx/s with 50 nodes. Narwhal uses certificates of availability and worker-based scaling to achieve high throughput. MESH-BFT operates per-record rather than batching into blocks, sacrificing raw throughput for per-record granularity, and adds diversity and post-quantum models absent from Narwhal/Tusk.

Aleph BFT [6]. Gagol et al. designed a leaderless protocol with virtual voting derived from DAG structure and binary agreement via dual-threshold coin toss. Aleph is architecturally closest to MESH-BFT (leaderless, DAG-based, zero extra messages for ordering) but produces total order. It has no diversity model and no post-quantum analysis.

Bullshark [13]. Spiegelman et al. made DAG BFT practical by simplifying DAG-Rider's ordering rule while maintaining the same theoretical properties. Bullshark achieves 130,000 tx/s with sub-2-second finality on 50 nodes. Like all DAG-based protocols, it assumes independent Byzantine failures and requires total ordering.

7.3 Probabilistic Consensus

Avalanche [3]. Team Rocket introduced metastability-based consensus via repeated subsampled voting (sample size k=20). Avalanche is the closest system in spirit to MESH-BFT: it supports partial ordering, and independent transactions finalize in parallel without competing for consensus bandwidth. However, Avalanche provides only probabilistic safety (never absolute deterministic guarantees), uses sub-sampling rather than attestation accumulation, and has no diversity or post-quantum model. MESH-BFT provides deterministic safety with diversity weighting.

7.4 Per-Object Finality

Sui Lutris [7]. Blackshear et al. combined Byzantine Consistent Broadcast (for owned-object transactions requiring only 2f+1 acknowledgements and sub-second finality) with full Mysticeti consensus (for shared-object transactions). Sui Lutris achieves 300,000 tx/s for owned objects on 100 validators. The single-writer model is narrower than MESH-BFT's general DAG mesh, which supports multi-parent records. MESH-BFT additionally provides diversity weighting and post-quantum security.

FastPay [8]. Baudet et al. designed high-performance settlement for pre-funded payments using Byzantine Consistent Broadcast with O(N) messages and single round-trip confirmation. FastPay is a predecessor to Sui with a much narrower scope (simple payment transfers only). MESH-BFT generalizes per-record finality beyond the payment use case.

7.5 Validator Diversity

GPoS [9]. The Geospatially-aware Proof of Stake protocol combines a Geographic Diversity Index (GDI) with stake to compute voting power, tested on HotStuff and CometBFT with a 45% improvement in Gini coefficient. GPoS is the closest prior art to MESH-BFT's diversity model, but it provides no formal BFT safety proof incorporating diversity as a parameter — the GDI is

evaluated empirically only. MESH-BFT formalizes diversity in the safety proof itself (Theorem 1).

Ethereum Validator Correlation Analysis [10]. Brown and Bautista-Gomez developed a correlation-adjusted index for measuring validator decentralization in Ethereum. This is a measurement and analysis contribution, not a consensus protocol modification. MESH-BFT operationalizes similar correlation insights directly in the consensus mechanism.

Flexible BFT [11]. Malkhi et al. introduced heterogeneous trust assumptions where clients select their own resilience model. "Diversity" in Flexible BFT refers to trust assumption diversity, not validator attribute diversity. MESH-BFT addresses a different dimension: the correlation structure of validator infrastructure.

Ethereum Client Diversity [14]. The Ethereum Foundation's client diversity initiative addresses the engineering concern that validator software monoculture creates correlated failure risk. Ethereum incorporates diversity heuristically in slashing penalties but has no formal model. MESH-BFT formalizes this concern with Definitions 11-15 and proves its safety implications.

7.6 Post-Quantum Blockchain

SoK: Blockchain Consensus in the Quantum Age [12]. Nash provides the most comprehensive survey of quantum-resistant consensus approaches (2019-2024). The survey catalogs existing defenses but does not propose a new adversary model — it identifies the gap that MESH-BFT fills with the QPT adversary (Definition 22) and dual-algorithm security proof (Theorem 2).

QDBFT [17]. QDBFT uses Quantum Key Distribution (QKD) for BFT, requiring quantum hardware (QKD networks) rather than post-quantum classical cryptography. This is a fundamentally different approach — MESH-BFT uses classical post-quantum algorithms (ML-DSA-65, SLH-DSA) deployable on existing hardware.

---

8. Discussion and Future Work

8.1 Limitations

Testnet scale. Our empirical results come from a 5-node testnet. While the measured gossip latency (p50=250ms) and conservation invariant validation are informative, claims about behavior at 1,000+ nodes remain theoretical projections based on the sqrt(n) gossip model. Large-scale simulation is needed to validate these projections.

Simulation status. The safety-under-adversarial-conditions Monte Carlo *has* been executed: the `tools/mesh-bft-sim` harness sweeps a 64-combination adversary grid (network sizes {10, 50, 100, 500} × Byzantine fraction {0.10, 0.20, 0.30, 0.33} × sybil-cluster size {1, 5, 10, 50}) at the spec parameterization $\gamma = 0.1$ and records **zero safety violations** (committed `results.json`; re-validation at the shipped $\gamma = 0.2$ is pending). Liveness under churn and throughput under sustained load (Section 6.3) remain analytically modeled, not yet simulated.

ZK curve limitation. The *specified* ZK proof subsystem targets BN254 (Groth16), which is not post-quantum secure — the bilinear pairing is vulnerable to quantum attacks via Shor's algorithm on the discrete log problem. The shipped Phase-1 construction is SHA3-256 hiding commitments (hash-based, no curve exposure; the Groth16 wire version is rejected fail-closed until a prover/verifier lands), so this limitation becomes live when the zk-SNARK layer ships. Lattice-based ZK-SNARK constructions are an active research area; migration is planned when suitable schemes mature.

Profile B weakness. Profile B (single-signature, ML-DSA-65 only) is strictly weaker than Profile A under quantum attack. Browser and constrained-device nodes using Profile B create a heterogeneous security landscape. The mitigation (requiring minimum Profile A quorum for settlement) reduces this risk but does not eliminate it.

Diversity oracle. The witness profile P(I) = (org, subnet, geo) relies on self-reported or inferred attributes. An adversary that can convincingly fake diverse profiles can circumvent the diversity discount. This is a bootstrapping problem: early networks with few validators have limited ability to verify profile claims. As the network grows, cross-validation and reputation mechanisms can harden the oracle.

Single-network scope. All definitions and proofs in this paper assume one network with one stake universe: the settlement threshold (Definition 18) is computed against a global S_total, and the safety argument (Theorem 1) bounds adversarial stake relative to that single supply. Deployments that run multiple networks with disjoint roots of trust (e.g., permissioned or air-gapped instances) are sound only as **independent instances** of MESH-BFT — each with its own stake universe, settlement threshold, and security perimeter; the theorems make no claim across instances. Two consequences follow. (1) **No external time:** the protocol attests causal order; timestamps are operator-supplied and epochs use network-relative timeouts, so nothing in this paper binds records to calendar time — externally-anchored state commitments are required wherever wall-clock claims matter. (2) **Cross-network history transfer is out of scope:** merging two long-disjoint DAGs (e.g., a private network publishing its history into a public one) has no conflict-resolution or finality-preservation semantics in this model. Until such semantics are formalized, the sound deployment rule is publication-as-attested-bundle: the receiving network attests the existence, identity continuity, and anchor trail of the imported bundle, but does not extend its own settlement guarantees over the bundle's internal records, and imported history confers no stake, witness eligibility, or trust standing in the receiving network (see Protocol Whitepaper Section 10.6.3).

8.2 Future Work

Formal verification. Partial machine-checked verification is already in place and CI-gated: TLA+ specifications (`spec/tla/`) exhaustively model-check the core safety and settlement-liveness properties on bounded instances (the `formal-tla` CI job), and a ProVerif model (`spec/proverif/`, twenty-five scenarios) verifies the post-quantum handshake's secrecy and authentication properties (the `formal-proverif` CI job). What remains future work is theorem-level machine-checked verification of the diversity-weighted settlement bound (unbounded rather than bounded-instance) — a Lean 4 development is the intended next step.

Multi-network semantics. Formalizing cross-network history transfer: a merge model for long-disjoint DAGs with explicit conflict-resolution rules, finality-preservation statements across the publication boundary, and a safety theorem for the publication-as-attested-bundle rule described in Section 8.1. This is the consensus-side counterpart of the NETWORK_PUBLISH protocol (Protocol Whitepaper Section 10.6.3).

MPC ceremony for Groth16. The *specified* Groth16 subsystem requires a trusted setup (the shipped Phase-1 SHA3-256 commitments require none). When the prover/verifier lands, a multi-party computation ceremony with public verifiability would eliminate the trusted setup assumption.

Interval Tree Clocks. The current timestamp function tau uses f64 timestamps, which are adequate for ordering but imprecise for causal tracking in distributed settings. Interval Tree Clocks [14] provide a more principled approach to causal ordering in dynamic distributed systems.

Lattice-based ZK-SNARKs. When post-quantum ZK proof systems (e.g., based on Module-LWE or hash-based commitments) reach practical maturity, adopting them in place of the specified BN254 Groth16 target will keep the proof layer fully post-quantum as it moves beyond the

Phase-1 hash commitments.

Large-scale simulation. A Monte Carlo simulator covering 10,000-100,000 nodes under varying adversarial conditions (Byzantine fraction, sybil concentration, network partitions) is needed to validate the scaling projections and identify potential failure modes not captured by the analytical model.

Communication complexity. The sqrt(n) gossip model gives propagation bounds, but a formal analysis of total communication complexity for attestation accumulation — including bandwidth per node and message amplification — remains open.

References

[1] M. Castro and B. Liskov, “Practical Byzantine Fault Tolerance,” OSDI 1999.

[2] M. Yin et al., “HotStuff: BFT Consensus with Linearity and Responsiveness,” PODC 2019.

[3] Team Rocket, “Scalable and Probabilistic Leaderless BFT Consensus through Metastability,” 2020.

[4] I. Keidar et al., “All You Need Is DAG,” PODC 2021.

[5] G. Danezis et al., “Narwhal and Tusk: A DAG-based Mempool and Efficient BFT Consensus,” EuroSys 2022.

[6] A. Gagol et al., “Aleph: Efficient Atomic Broadcast in Asynchronous Networks with Byzantine Nodes,” AFT 2019.

[7] S. Blackshear et al., “Sui Lutris: A Blockchain Combining Broadcast and Consensus,” CCS 2024.

[8] M. Baudet et al., “FastPay: High-Performance Byzantine Fault Tolerant Settlement,” AFT 2020.

[9] “GPoS: Geospatially-aware Proof of Stake,” arXiv:2511.02034, 2025.

[10] A. Brown and L. Bautista-Gomez, “Exploring Correlation Patterns in the Ethereum Validator Network,” arXiv:2404.02164, 2024.

[11] D. Malkhi et al., “Flexible BFT,” CCS 2019.

[12] L. Nash, “SoK: Blockchain Consensus in the Quantum Age,” IACR ePrint 2025/1564.

[13] A. Spiegelman et al., “Bullshark: DAG BFT Protocols Made Practical,” CCS 2022.

[14] Ethereum Foundation, “Client Diversity,” ethereum.org/developers/docs/nodes-and-clients/client-diversity.

[15] NIST, “FIPS 204: Module-Lattice-Based Digital Signature Standard (ML-DSA),” 2024.

[16] E. Buchman, “Tendermint: Byzantine Fault Tolerance in the Age of Blockchains,” 2016.

[17] “QDBFT: Dynamic Consensus for Quantum-Secured Blockchain,” arXiv:2602.11606, 2026.