Open replica sets + Bitcoin-style halving rewards

Date: 2026-05-25 **Status:** Design proposals from James 2026-05-25 — (1) minimum 5 replicas at launch but unbounded growth as the system matures; (2) Bitcoin-style halving schedule for replica token rewards. This document analyzes both, surfaces the structural constraints, and proposes a concrete design that delivers what James actually wants without breaking the protocol's trust model. **Scope:** Plain-English explanation of both proposals + the trust-model constraint that makes "infinite replicas" non-trivial + the Cosmos-style "bonded pool with active signing set" pattern + halving mechanics + supply math + how this revises Option III from `replicas-detail-2026-05-25.md` §11.

> ■■ ***Alignment note (2026-06-28) — read first.*** *This is a dated iteration record, superseded in framing by `replica-system-2026-05-25.md` and the whitepaper (§9, §11). Canonical source for changes: `DECISIONS.md` (D-013). -* ***The 5,000 XMR mint-fee cap was removed (D-013).*** *Post-cap reward analysis is moot; the XMR fee share now flows continuously. -* ***Epoch is now two-cadence:*** *the ~1-hour epoch in this doc's halving math is the* ***attestation epoch*** *(emission cadence), now formally distinct from the 7-day* ***registry epoch*** *(snapshots). Under that reading the reward math here — R■ = 1,000/epoch, 8,760 attestation epochs/year, 8.76M/replica/year, cumulative ~17.52M — remains current.*

---

0. The two proposals in James's words

1. **"Minimum 5 to start but allow an infinite amount of them — so more people can get involved as the system grows."**
2. **"A halving system for replicas, like Bitcoin? Running replicas gives you X tokens as a reward + XMR. Every X months/years there's a halving, reducing the reward and therefore contributing to reduction of emissions."**

Both proposals reflect good intuitions about how production token protocols actually work over time. Halving is straightforwardly excellent — recommended for adoption. Open replica sets need a small refinement to preserve security — the design is a "bonded permissionless pool with a Foundation-governed active signing set," which is the standard pattern in Cosmos, Polkadot, and similar protocols.

Part 1 — Plain English

---

1. Why "infinite replicas" needs refinement

The current 5-replica design works because **clients know exactly which 5 replicas to trust** — the SDK ships with 5 specific public keys, and clients verify that 3 of those specific keys signed any snapshot they accept. The SDK is shipped, signed, and reproducibly built; clients can't be tricked into trusting a fake replica.

If anyone can become a replica, then a hostile party can spin up 100 fake replica processes overnight, sign whatever they want, and the protocol has no built-in way to tell clients "trust replicas A, B, C, D, E but not these 100 you've never heard of." The trust model collapses.

The refined design that delivers what James wants:

Anyone can bond and become a candidate replica. Bonding is permissionless. You acquire tokens, post them as collateral, and run replica software. You're in the protocol's "bonded pool."

The 5 (or more later) replicas that count for the quorum are the "active signing set." This is a subset of the bonded pool, decided either by Foundation governance (early days) or by automatic protocol rules (later — top-N by bond stake, similar to Cosmos validator set selection). Clients pin the active signing set, not the bonded pool.

Active signing set grows over time. At v1 launch: 5 active. As the protocol matures and demand for the registry grows, the active set scales: 5 → 7 → 9 → 11 → ... → eventually arbitrary N. Each expansion is a Foundation governance vote in v1; in v2 it can become automatic (top-N by bond).

Bonded-but-inactive replicas wait in line. They earn a smaller reward for standing ready as backup; if an active replica drops out or gets slashed, the highest-bonded inactive replica is automatically promoted to fill the slot.

The 3-of-N quorum threshold scales with the set. 3-of-5 at launch. 3-of-7 when set grows to 7. 5-of-9 at 9. 7-of-13 at 13. The threshold is always at least the strict majority of the active set, so the BFT-tolerance property holds. (Concrete formula in §7.)

This is exactly the Cosmos approach — anyone can stake to a validator, but only top 100 by stake are in the active set. Polkadot has the same structure. Hyperliquid uses a smaller version (top 24 by stake; what we now call the "active validator set" was capped explicitly).

The result is functionally what James asked for: as the system grows, more people can run replicas and earn rewards; the protocol becomes more decentralized over time; nothing in the design caps participation; the only constraint is that **the act of joining the active signing set is governed**, not the act of running replica software.

2. Halving rewards — strongly recommended

The halving idea is excellent. It replaces the previous "linear decay over 36 months" emission schedule from `replicas-detail-2026-05-25.md` §11 with something cleaner, more predictable, and more recognizable to anyone familiar with Bitcoin.

How it works:

- **Launch reward:** at v1 launch, each replica earns N tokens per epoch for participating in signing.
- **Halving event:** every 12 months, the per-epoch reward halves.
- **Continues indefinitely:** there's no terminal "rewards go to zero" — the math just asymptotes. After 4 halvings the reward is 1/16 of launch; after 8, 1/256; after 16, ~1/65,000. By that point operators rely on XMR fee share + token appreciation, exactly like Bitcoin miners rely on transaction fees as block subsidies shrink.

What this buys:

- **Predictable supply curve.** Total tokens emitted to replicas asymptotically approaches 2× the first-year emission. Easy to model; no surprise inflation; no governance dependency for the schedule.
- **Operator urgency.** Replicas joining at launch earn the most; operators joining at year 3 know they're at 1/8 the reward. Creates natural early-mover incentive.
- **Replaces the "sunset clause."** The prior design had a sunset that auto-halted emissions if marketplace burn was low. Halving doesn't need that — emissions naturally taper without needing a "is the marketplace working?" check.
- **Marketing narrative.** "Halving" is a well-understood concept in crypto. Easier to communicate to potential operators than "linear decay over 36 months."
- **Pairs cleanly with the hybrid reward model.** From `replicas-detail-2026-05-25.md` §11 the recommended Option III splits replica reward between token emissions and XMR fee share. Halving only applies to the token side; XMR fee share doesn't halve (it's revenue-tracking, not emission-tracking).

3. How the two proposals fit together

The open replica set and halving combine cleanly. Consider the design 4 years in:

- **Active signing set:** grown from 5 → 9 via two Foundation-governance expansions at year 1 and year 3.
- **Bonded pool:** 25 candidate operators bonded, 9 currently active, 16 waiting as standby.
- **Token reward per epoch per replica:** at 1/16 of launch level (4 halvings = 24 months / 12 = 2 halvings... actually let me recompute — 4 years / 12-month halving = 4 halvings, so 1/16). Each active replica's token earnings have dropped sharply; XMR fee share has stayed roughly flat (depending on registration volume).
- **Operator economics:** rewards in token terms are smaller per epoch, but the protocol's expected token price has grown with the network. Halving creates structural scarcity that supports price; operators earn fewer tokens but each is worth more.

The two designs reinforce each other. Open participation lets more operators join over time, but halving controls supply so the per-operator economics don't get diluted by set growth.

4. A worked example

Alice is one of the original 5 replica operators at v1 launch in year 0.

- **Year 0:** Alice bonds 100,000 tokens, runs her replica, earns 1000 tokens/epoch + small XMR share. At 8760 epochs/year, that's 8.76M tokens/year + ~5,000 XMR-equivalent.

- **Year 1 (halving 1):** Per-epoch reward halves to 500 tokens. Alice now earns 4.38M tokens/year. XMR share grows slightly as registration volume grows.
- **Year 2 (halving 2):** Reward at 250 tokens/epoch. 2.19M tokens/year. XMR continues to grow.
- **Year 3:** Foundation governance votes to expand active set from 5 → 7. Two operators are promoted from the bonded pool. Each replica still earns 250 tokens/epoch (the per-replica rate, not the total), but the total emission per epoch is now 7 × 250 = 1750 (up from 5 × 250 = 1250). This is a deliberate design call — incentivizing growth rewards new operators.

Wei the hosting-company operator joins the bonded pool in year 2 with two replicas. He's in standby until year 3 when the expansion happens; one of his replicas gets promoted, the other remains in the pool waiting for a future expansion. From year 3 onward Wei earns the same per-epoch reward as Alice.

Year 4 (halving 3): reward at 125 tokens/epoch. Year 5: 62.5. Year 6: 31.25. By year 6 token emissions are at 1/32 of launch; operators are running mostly for XMR share + token appreciation; if the network has actually grown the way the design hopes, those two factors more than compensate.

5. What this means for current parameters

Three things change from the prior docs:

1. **Replica set is not capped at 5.** It starts at 5, grows by governance vote, no hard ceiling.
2. **Bond is permissionless.** Anyone can bond; only the top-of-pool wait for active-set slots.
3. **Halving replaces linear decay.** 12-month halving schedule replaces the "linear decay over 36 months" from `replicas-detail-2026-05-25.md` §11.

Two things stay the same:

- **Unbond cooling: 90 days** (per James 2026-05-25, from `replicas-detail-2026-05-25.md` §12.3).
- **Hybrid reward model (Option III)** from `replicas-detail-2026-05-25.md` §11 — but the token-side emission curve is now halving-shaped, not linear-decay-shaped.

Part 2 — Technical detail

---

6. Why open replica sets are non-trivial — the trust-model constraint

The 3-of-5 quorum design works because clients have a **pinned, fixed, known-in-advance set** of 5 public keys. The SDK ships with these keys; clients verify that 3 of those 5 specific keys signed any snapshot they accept.

If we relax "fixed" to "anyone can be a replica," we need a different mechanism for clients to know which signatures count toward quorum. Three approaches with concrete trade-offs:

6.1 Approach 1: client queries the protocol for the current active set

Mechanism: an on-chain registry contract (or equivalent) lists current active replicas. Clients query this registry at startup (or on cache invalidation) to learn the current 5 / 7 / 9 / N replicas to trust.

Pros: fully dynamic; supports arbitrary set growth. Cons: requires clients to query the registry chain (new dependency); creates a bootstrapping problem (how does the client know the address of the registry contract?); adds a centralization point (the registry contract is its own trust anchor).

6.2 Approach 2: SDK-pinned active set, updated per SDK release

Mechanism: the SDK release artifact ships with the current active set. Set expansions happen at SDK release boundaries; clients update SDKs to get new pubkeys.

Pros: no new infrastructure dependency; SDK is already a trust anchor via reproducible build + signature. Cons: set updates are tied to SDK release cadence; old SDK clients keep using the old set; emergency replica additions require emergency SDK releases.

6.3 Approach 3: Foundation-signed active-set manifest, fetched and verified by clients

Mechanism: Foundation publishes a signed "active-set manifest" containing the current pubkeys. Clients fetch it (cached, signed by Foundation key) periodically. Updates can happen without SDK releases.

Pros: dynamic without SDK release dependency; Foundation key is already a trust anchor. Cons: Foundation signature becomes load-bearing; if Foundation key is compromised, attacker can substitute active set.

6.4 Recommendation for v1: Approach 2 with v2 graduation to Approach 1

v1 ships with Approach 2 (SDK-pinned active set). Initially 5 pubkeys; SDK releases at predictable intervals (every 3-6 months) carry new pubkeys when the active set expands.

v2 introduces Approach 1 (on-chain registry). The token's settlement chain hosts a registry contract; bonded operators register; top-N-by-bond automatically become the active set; clients query the chain.

This phasing matches the legitimate maturity of the protocol: v1 launches with a simpler trust model that's easier to audit and reason about; v2 unlocks the truly permissionless dynamic set once the token economy is established and the on-chain query mechanism can be relied on.

7. The recommended design — bonded permissionless pool with Foundation-governed active set

7.1 The two-tier structure

- **Tier 1: Bonded pool.** Permissionless. Anyone can bond N tokens and run replica software. Their replica process syncs with the network via gossip; they receive submissions, run anti-entropy, and stand ready to sign. They earn a small "bonded-but-inactive" reward (provisionally 10% of an active replica's per-epoch token reward) for standing ready.
- **Tier 2: Active signing set.** A subset of Tier 1. These replicas' signatures count toward the 3-of-N quorum. They earn the full per-epoch token reward + XMR fee share.

7.2 How operators move from Tier 1 to Tier 2

In v1 (Foundation-governed): - New active-set slots open when Foundation governance votes to expand the set. - Foundation selects from the bonded pool, weighted by: bond size, operating history (uptime, gossip-convergence speed, no slashing events), jurisdictional diversity, hosting diversity. - Existing active replicas remain unless they exit or are slashed.

In v2 (auto-selected by bond): - Top-N-by-bond from Tier 1 are automatically Tier 2. - Set size N is determined by Foundation-governed protocol parameter (can grow over time without changing the selection mechanism). - Top-N changes happen at epoch boundaries; transition is rate-limited (max 1 replica change per epoch) to prevent churn.

7.3 Active-set size N — growth schedule

Provisional governance-vote-triggered schedule:

Year	Active set N	Quorum threshold	Why this milestone
0 (v1 launch)	5	3-of-5	Minimum recruitable + auditable
1	5	3-of-5	No expansion in year 1; let v1 stabilize
2	7	4-of-7	Marketplace activity establishing; expansion incentivizes new operators
3	9	5-of-9	Maturing operator pool; halving 3 makes the per-operator economics less attractive, expansion compensates
5	13	7-of-13	v2 likely available by this point; transition to automatic top-N selection
7+	governance-set	strict majority	v2 fully online; growth becomes automatic

The pattern: **N is odd** (so the strict-majority quorum is unambiguous), **N grows by 2 at a time** (one Foundation-side, one community-side balance can be preserved), and the **quorum is always strict majority** (so BFT-tolerance holds — strictly less than half can be Byzantine without forging snapshots).

7.4 Why strict-majority quorum

In the 3-of-5 case, an attacker needs 3 of 5 replicas to forge a snapshot — strict majority. For 7, that's 4-of-7; for 9, 5-of-9. The general formula: `quorum = floor(N/2) + 1`.

This is the simplest defensible threshold. Could be tightened (e.g., 5-of-9 → 6-of-9 = supermajority) for higher trust requirements, but at the cost of liveness (more replicas need to be online).

7.5 What about really, really large N?

There's no protocol-imposed ceiling on N. But practically:

- Each active replica adds a signature to every snapshot. Snapshot size grows linearly with N.
- Each active replica is a target for compromise. The attack surface grows linearly.
- Each active replica must do gossip reconciliation with N-1 peers. Gossip cost grows with N²ish.

In practice, the protocol probably plateaus around N=25-50 — beyond that, the operational overhead exceeds the security benefit. But this plateau is a design choice + can be revisited; it's not a protocol limit.

8. Halving mechanics

8.1 The schedule

- **Per-replica per-epoch token reward at v1 launch:** R■ (provisional: 1000 tokens).
- **Halving period:** 12 months (provisional). Can be 6, 12, 18, or 24 months — see §9.4 for analysis.
- **At each halving:** R_n = R_{n-1} / 2.
- **Halving 1:** R■ = 500 tokens/epoch. Halving 2: R■ = 250. Halving 3: R■ = 125. Halving 4: R■ = 62.5.
- **No floor.** The reward asymptotes to zero; operators rely on XMR fee share + token appreciation as halvings progress.

8.2 Set-size interaction

Two design choices for how halving interacts with active-set growth:

Choice A: per-replica reward halves; total emission grows with N. - Each active replica earns R_n per epoch regardless of set size. - Total emission per epoch = N × R_n. - When the set grows from 5 → 7, total emission jumps 1.4× even without a halving. - Pros: new operators join at the same per-replica reward as existing operators; incentivizes growth. - Cons: emission rate is partially controlled by governance (set-size decisions), not just halving.

Choice B: total per-epoch reward halves; per-replica drops with N. - Total emission per epoch is fixed: T_n = R■_total / (2^n). - Per-replica reward = T_n / N. Drops as N grows. - Pros: total emission rate is fully determined by halving; no surprise from set-size growth. - Cons: existing operators see their per-epoch reward drop when the set expands; mild disincentive for advocating expansion.

Recommended: Choice A. Per-replica predictability matters more than total-emission predictability. Set-size growth happens slowly via Foundation governance; the additional emission from N going 5 → 7 is bounded (1.4×). Halvings provide the long-term emission control.

8.3 Concrete supply math (Choice A)

Per-replica per-year emission (8760 epochs/year):

- Year 0-1: 1000 × 8760 = 8.76M tokens
- Year 1-2: 500 × 8760 = 4.38M
- Year 2-3: 250 × 8760 = 2.19M
- Year 3-4: 125 × 8760 = 1.10M
- Year 4-5: 62.5 × 8760 = 547,500
- ...
- Cumulative per-replica over all halvings: 1000 × 8760 × 2 = ~17.52M tokens

If active set grows on the schedule in §7.3: - Year 0-1: 5 replicas × 8.76M = 43.8M tokens - Year 1-2: 5 × 4.38M = 21.9M - Year 2-3: 7 × 2.19M = 15.3M - Year 3-4: 9 × 1.10M = 9.9M - Year 4-5: 9 × 547,500 = 4.93M - Year 5-6: 13 × 273,750 = 3.56M - Year 6-7: 13 × 136,875 = 1.78M - Year 7-8: 13 × 68,438 = 890k - ...continues halving forever

Total cumulative emission to active replicas: ~110M-130M tokens across infinite halvings (depending on exact set-growth path).

Plus bonded-but-inactive pool: ~10% of active replicas' rate per pool member. If pool size is ~20-30 members average, that's ~10-15% additional emission. Total: ~125M-150M tokens to replicas across the lifetime of the protocol.

As a percentage of supply: if total supply is 1B (Hyperliquid-equivalent), replica emissions are ~12-15% of supply. Allocated within the "Operators" bucket from `tokenomics-option-e-detail-2026-05-25.md` §11 (provisional 25-35%). The remaining operator allocation funds mint authority compensation + bonded-but-inactive pool rewards + bootstrap subsidies.

8.4 What about halving periods other than 12 months?

Halving period	Years to 1/16 reward	Years to 1/256 reward	Notes
6 months	2	4	Aggressive; operators see economics evolve fast; may discourage long-term commitment
12 months (recommended)	4	8	Balanced; matches typical operator cost-recovery horizons
18 months	6	12	Conservative; emissions stay meaningful longer; more cumulative supply
24 months	8	16	Very conservative; emissions stay material for a decade+; significantly more cumulative supply
48 months (Bitcoin)	16	32	Bitcoin's choice; appropriate for a 100-year asset; probably too slow for a new protocol launching now

Recommendation: **12 months.** Fast enough to demonstrate the halving narrative within the first 12 months of v1; slow enough that operators have meaningful time horizons; matches the "halve every year" mental model that's easy to communicate.

8.5 First halving — choosing the moment

Bitcoin halvings happen on a block-count basis (every 210,000 blocks ≈ 4 years given 10-minute blocks). For Nightshade, the analogous trigger is an **epoch count**: with 1 epoch = 1 hour, 8760 epochs = 1 year. Halvings trigger at epochs 8760, 17520, 26280, etc.

Alternatives: - **Wall-clock date.** Halving happens on a calendar date. Pros: predictable for operators. Cons: epoch count drifts if epoch duration is ever changed via governance. - **Total emission count.** Halving when N tokens have been emitted (Bitcoin's mechanism in spirit). Pros: tied to actual supply. Cons: complex math for operators; subject to set-size variation.

Recommendation: **epoch-count trigger.** Matches the protocol's natural cadence; aligns with how snapshots are constructed; resistant to wall-clock manipulation.

9. Open design questions specific to this proposal

1. **Halving period choice — final.** Recommendation 12 months; James can override with 6, 18, or 24.
2. **Initial per-replica per-epoch reward R■.** Provisional 1000; depends on launch token price + target operator USD economics. Calibrate via the strategic-opportunity discussion.
3. **Active-set growth path.** §7.3 sketch is 5 → 5 → 7 → 9 → 13 → governance. Actual cadence depends on demand + operator pool depth.
4. **Bonded-pool reward rate.** Provisional 10% of active rate. Could be 5% (less attractive to standby; cheaper for protocol) or 20% (more attractive).
5. **Bonded-pool size cap.** Should there be a maximum pool size, or is it unlimited? Unlimited risks emissions ballooning if many operators bond just for the standby reward.
6. **Active-set selection criteria in v1 Foundation phase.** §7.2 lists provisional criteria; needs concrete operator-onboarding doc.
7. **v2 transition mechanics.** When does the Approach 1 (on-chain registry) replace Approach 2 (SDK-pinned)? What triggers the v2 release?
8. **Multi-bond per operator?** Can one operator (one legal entity) run multiple replicas under different signing keys, each with its own bond? Affects Sybil-resistance design.
9. **Cross-replica delegation.** Should bonded-but-inactive operators be able to delegate their bond to active operators (Cosmos-style)? Adds complexity but enables aggregator-style participation.
10. **Reward smoothing.** Halvings are step functions. Should the per-epoch reward decay smoothly between halving events (e.g., 1/12 reduction per month) instead of dropping abruptly? Bitcoin uses step function; some other protocols smooth.

10. Revised hybrid reward model (Option III, halving-aware)

Updates to `replicas-detail-2026-05-25.md` §11.3 Option III:

10.1 Token emissions (revised)

- At v1 launch: per-replica per-epoch reward = 1000 token (R■).
- Halving every 12 months: R_n = R■ / 2^n.
- No floor; emissions asymptote to zero.
- Choice A interaction with set size: per-replica reward stays constant within a halving period regardless of how many replicas are in the active set; total emission grows with N.
- Bonded-but-inactive pool: each member earns 10% of active-replica rate per epoch.

10.2 XMR fee share (unchanged)

- 1% of each agent-registration mint fee (0.027182 XMR) → replica reward pool.
- Per-registration replica reward = 0.000272 XMR.
- Pool split per-registration among the replicas that signed the snapshot containing that registration.
- Continues at constant rate (no halving — this is real revenue, not emission).
- Bounded by the 5,000 XMR cap; post-cap, XMR side of reward stops accumulating new revenue. Halving-based token side continues.

10.3 Bond requirements (unchanged from prior doc except set-size implications)

- Bond per replica: 100,000 token at launch.
- Bond can be adjusted via Foundation governance to maintain USD-equivalent stability.
- 90-day unbond cooling (per James 2026-05-25).
- Slashing conditions unchanged (§12 of `replicas-detail-2026-05-25.md`).
- Bonded-but-inactive operators: same bond requirement (100k tokens), same slashing terms during standby.

10.4 Slashing destination

Burned, not redistributed. Same as prior doc.

---

11. Summary

Companion to `tokenomics-research-2026-05-25.md` (wide research), `tokenomics-option-e-detail-2026-05-25.md` (Option E first-pass), and `replicas-detail-2026-05-25.md` (replica role + reward design first-pass). Together these four documents form the working tokenomics design surface as of 2026-05-25. None is a binding decision; all four together describe the design in enough depth for counsel review + strategic-opportunity-driven final calibration.