XI'AN · HYPERSCALE FOR RADIX

Hyperscale Weekly

Split the work, not the world

WEEK 40 · 28 SEPT 2026

28 Sept – 4 Oct 2026

CURRENT MILESTONE

Milestone 2 · Radix Engine

Porting the Radix Engine so Scrypto runs across shards

Week 9 · since August 2026

Previously: flightofthefox comprehensively rebuilt the crossing system for tracking cross-shard transactions, and flagged local execution as a major upcoming challenge.

This week, flightofthefox turned from last week's crossing-system rebuild toward a different kind of hard problem: what happens when a shard forks, halts, or needs a new validator seated mid-flight. The commits tell a story of making recovery and validator seating work cleanly on running shards — both in production and in the simulator.

19	+4.0k −1.1k	64	0
-89% vs last wk	-84% vs last wk		
COMMITS	LINES CHANGED	FILES TOUCHED	DAY COMMIT STREAK

COMMITS / DAY



peak 19/day · 1/7 days active

WHAT FLIGHTOFTHEFOX BUILT

Rebuilding forked shards from anchors

flightofthefox made forked production and simulator shards rebuild themselves from a known anchor point before validators are seated, ensuring recovery always starts from verified state rather than untrusted data.

'see "The Archive" on hyperscale.rs

Seating validators on live shards

New validators can now join a running shard once their local store is settled, and co-hosted validators can be seated and released on both production and sim shards — the machinery for the trickle rotation that keeps committees honest.

'see "The Lottery" on hyperscale.rs

Halt recovery gets robust

Recovery bindings are now resolved from bounded history rather than pending records, halting shards freeze by withholding validators instead of cutting hosts, and the recovery floor is set before discarding old frontier data.

'see "The Overlap" on hyperscale.rs

THE BOTTOM LINE

STANDOUT CHANGE

Validators can now be seated into and released from running shards — the live machinery for committee rotation.

WHAT'S NEXT

Likely next: continuing to harden halt recovery and connect the validator-seating path to the trickle rotation system.

Written by AI from public commits, the community chat and hyperscale.rs — may contain mistakes.

From the community

What people are saying, and the words behind the work.

HEARD IN THE CHAT

The mood: The chat was dominated by a heated debate about whether XRD holders benefit from Hyperscale, with flightofthefox firmly declining to promise anything beyond the tech.

> *"what XRD does is not up to me. i'm just building tech, if radix wants to give a grant because it helps achieve their goals than great. this is a good alignment of incentives"*
>
> — FLIGHTOFTHEFOX, IN THE COMMUNITY TELEGRAM

FLIGHTOFTHEFOX EXPLAINED

DePIN on Hyperscale

DePIN doesn't pose technical challenges — the on-chain part is usually a tiny fraction of the architecture, and the smart contracts platform itself is a sufficient framework.

Token for DePIN rewards

You're better off with discrete tokens for pure-play services rather than muddying up tokenomics that exist to fulfill concrete security goals.

His role vs XRD price

He builds technology and can't solve every problem; whether XRD benefits is outside his domain, though he works hard to be competitive with the market.

Keeping the channel open

Without Radix there'd be no Telegram channel; he keeps it open to share enthusiasm for consensus and answer questions, because Radix people care about this stuff.

THIS WEEK'S CONCEPT

The Lottery — *A jury you can't bribe.* Seating and releasing co-hosted validators on running shards is the practical foundation for trickle rotation — the one-seat-at-a-time committee change that disperses corruption while keeping shards live.

hyperscale.rs/kleroterion

JARGON, DECODED

Anchor — A known-good checkpoint in a shard's history that recovery can safely rebuild from.

Co-hosted validator — Multiple validator identities running on the same physical machine, which is how vnodes work.

Frontier — The leading edge of a shard's chain that recovery must preserve before discarding.

Written by AI from public commits, the community chat and hyperscale.rs — may contain mistakes.

ZOOM OUT

The bigger picture

Where the work landed, and the map to read along on hyperscale.rs.

MOMENTUM · LINES CHANGED / WEEK



last 8 weeks · newest right

THIS WEEK'S WORK



WHERE THE WORK LANDED

node · The Lottery
validator seating and release on running shards

production · The Archive
forked shard rebuild from anchors before seating

simulation · The Crash Lab
halt recovery sims on co-hosted layouts

THE FULL PICTURE

The System
The whole sharded design at a glance.

The Journey
One transaction's trip through the network.

The Clock
Consensus-attested time across independent shards.

The Census
The leaderless beacon of validators, stake and shards.

The Overlap
Why two conflicting blocks can never both commit.

The Generals
All-or-nothing cross-shard commits, computed not voted.

The Archive
Every published byte is preserved or provably expired.

The Library
All state in one merkle tree; a shard is a subtree.

The Will
How in-flight transactions settle when a shard dies.

The Lottery
Random, ever-moving committees; proven cheats are jailed.

The Triage
Under overload, urgent traffic never waits for bulk.

The Governor
Validator entry is re-priced every epoch, like a market.

The Crash Lab
The simulator that replays any failure byte-for-byte.

The Proof
Proving safety with maths, not just tests.

The Asterisks
Every design's trade-offs — including Hyperscale's own.

THE ROAD TO MAINNET

M1	M2	M3	M4	M5	M6
Adaptive Sharding	Radix Engine	Gateway & API	Validator GUI	Migration	Live support
done	~5 mo	~3 mo	~3 mo	~3 mo	12 mo

Now: Milestone 2 · Radix Engine — Week 9 · since August 2026. Since day one: 5,133 commits — one developer, fully in public.

LINKS

WEBSITE
hyperscale.rs

DISCUSS THIS ISSUE ON X
x.com/i/status/2106972937819471965

CODE (PUBLIC)
github.com/hyperscalers/hyperscale-rs

MILESTONE PROPOSAL (PDF)
hyperscale.rs/xian_proposal.pdf

COMMUNITY (TELEGRAM)
t.me/hyperscale_rs

AUTHOR
flightofthefox

scan for hyperscale.rs

Questions? flightofthefox is always happy to discuss in the community Telegram — t.me/hyperscale_rs.

Written by AI from public commits, the community chat and hyperscale.rs — may contain mistakes.