XI'AN · HYPERSCALE FOR RADIX

WEEK 37 · 7 SEPT 2026

7–13 Sept 2026

Hyperscale Weekly

Split the work, not the world

CURRENT MILESTONE

Milestone 2 · Radix Engine

Porting the Radix Engine so Scrypto runs across shards

Week 6 · since August 2026

Previously: flightofthefox introduced the substate sweep for garbage collection and added signed network headers to transaction intents, with the Radix VM exploit prompting his XRD liquidation and a migration-path sketch

This was the biggest week of Milestone 2 so far — flightofthefox landed a near-complete rewrite of how the VM executes transactions, folding every path through a single checkpoint called the core boundary. Meanwhile, the cross-shard settlement system, the substate sweep, and shard-termination rules all matured in parallel, and the Telegram chat erupted into a heated debate about what Hyperscale means for Radix's future.

230	+19k −16k	559	0
same as last wk	+39% vs last wk		
COMMITS	LINES CHANGED	FILES TOUCHED	DAY COMMIT STREAK

COMMITS / DAY  peak 69/day · 5/7 days active

WHAT FLIGHTOFTHEFOX BUILT

Execution rewritten around one core

Every smart-contract execution path now runs through a single core boundary where modules are published, judged, and metered. The VM counts its own fuel per memory page, and code can only reach what it was explicitly lent — tightening security and aligning execution with sharding.

'see "The System" on hyperscale.rs

Cross-shard ledger takes shape

Abandonment records, counterpart questions, and state probes were unified into one ledger with a single retention rule, replacing scattered ad-hoc checks and making cross-shard commitments cleaner and more provable.

'see "The Generals" on hyperscale.rs

Sweep wired into block commits

The substate sweep was folded directly into the block commit path, and every transaction-derived artifact now carries a single expiry grace — so cleanup happens automatically as blocks finalize.

'see "The Archive" on hyperscale.rs

THE BOTTOM LINE

STANDOUT CHANGE

The VM execution model was rewritten around a single core boundary — the largest architectural change of Milestone 2 so far.

WHAT'S NEXT

Likely next: continuing to wire the settlement ledger and the sweep into the execution lifecycle, and pushing the VM core boundary toward feature parity with Babylon's component model.

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

More of what flightofthefox built

The week had more stories than one page could hold.

THE STORIES, CONTINUED

Shard death rules hardened

Rules for what happens when a shard terminates mid-transaction were tightened: the settled set and the fence now govern every straddler, with terminal evidence sized to the shard's final span.

'see "The Will" on hyperscale.rs

Fork safety tightened

Conflicting commits at the same block height are now refused outright, cross-chain claims must pass a single vote fence, and certificates from discarded rounds are evicted with their ticks — closing several fork-safety gaps.

'see "The Overlap" on hyperscale.rs

Sim and CI scaled up

The test suite now runs on arm64 runners with increased parallelism, the simulator gained placement-seated validators bound to their hosts, and both storage backends share one conformance test helper.

'see "The Crash Lab" on hyperscale.rs

ALSO THIS WEEK

- Memory store snapshots now share structure, reducing duplication across nodes.
- The beacon header window is bounded at insert, preventing unbounded growth.
- Merkle proofs carry empty siblings as single bits, saving wire space.
- Block abandonment records are now bounded in bytes rather than by count.
- RocksDB store cloning was simplified by dropping its wrapper type.
- Resharding split logic now leaves pool room the committee shuffle cannot take.
- Four reshape assertions were fixed to actually test what they claimed to test.
- Departed shards' stores are now served and routing is seeded at boot.

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: Heated debate dominated the chat about whether flightofthefox owes Radix a full migration, with frustration boiling over — but he reaffirmed he will help Radix adopt Hyperscale and noted the VM rewrit

> *“a grant is a beautiful trap: take fifty and wear the whole rap; if the flywheel won't spin they will ask "who cashed in?" - so i'll take the applause, not the clap”*
>
> — FLIGHTOFTHEFOX, IN THE COMMUNITY TELEGRAM

FLIGHTOFTHEFOX EXPLAINED

Why execution took longer

Leg local execution is an almost complete rewrite of how execution works, so the feature has taken longer to land than most.

Substate sweep vs GC

It’s cleaning up storage, not memory — reclaiming space from guards that don’t need to exist once validity windows for artifacts expire, like subintent nullifiers.

Session mode alternative

Shared-access components combined with subintents probably give as good a UX as sessions, without the potential foot-guns.

Migration harder than expected

He truly believed the existing Radix Engine would be more amenable to sharding — it was a rude awakening, and building a whole new VM was required.

THIS WEEK’S CONCEPT

The System — *Split the work, not the world.* The core-boundary rewrite is the largest architectural change of Milestone 2 so far — it redefines how every transaction executes, how resources are metered, and how code is contained, all to make execution compatible with sharding.

hyperscale.rs

JARGON, DECODED

Core boundary — The single checkpoint where all smart-contract code is published, validated, and metered before it can run.

Abandonment record — A note that a transaction left certain state behind and that state is now safe to reclaim.

Meter — A built-in counter inside each module that limits how much computation it can consume, preventing runaway code.

Vote fence — A rule that blocks a block from committing until its claims about other chains have been verified by a vote.

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



84% Rust · 14% Config · 2% Docs

WHERE THE WORK LANDED

types · The Generals
cross-shard ledger, settlement, and abandonment types

vm/sdk · The System
VM core boundary, metering, and reach limits

vm/harness · The Crash Lab
test harness, arm64 CI, conformance sharing

node · The Journey
node-level routing, fetch, and commit paths

vm/kernel · The System
kernel imports, core ABI, and fuel wiring

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 6 · since August 2026. Since day one: 4,464 commits — one developer, fully in public.

LINKS

WEBSITE
hyperscale.rs

DISCUSS THIS ISSUE ON X
x.com/i/status/2099362850812674372

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.