XI'AN · HYPERSCALE FOR RADIX

WEEK 41 · 5 OCT 2026
5–11 Oct 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 10 · since August 2026

Previously: flightofthefox had just rebuilt the crossing system, made forked shards recover from verified anchors, and got co-hosted validators seated into running shards.

Week 10 of Milestone 2 was a correctness blitz: 41 commits tightening how transactions hold locks, pay fees, and prove things to each other before their deadlines. In the chat, the community dug into the new VM's crypto plans, migration from Babylon, and a long, lively debate about who stores history.

41	+5.2k −1.6k	105	2
-76% vs last wk	-58% vs last wk		
COMMITS	LINES CHANGED	FILES TOUCHED	DAY COMMIT STREAK

COMMITS / DAY  peak 29/day · 4/7 days active

WHAT FLIGHTOFTHEFOX BUILT

Fees judged cell by cell

The VM now checks every debit against the fee holds standing on that exact account cell, and withholds credits from members that can settle without them. Fees get fairer and more precise as the engine takes shape.

'see "The Journey" on hyperscale.rs

Locks that can't deadlock

Dozens of fixes make two transactions circling the same state (a 'hold cycle') prove itself from readings taken before the deadline, then abort the later one. This is the cross-shard commit rule getting teeth.

'see "The Generals" on hyperscale.rs

Seating edge cases closed

Co-hosted validators can now be seated mid-flight, released once nothing awaits them, and loop-shared syncs follow the slowest seat. Committee rotation gets safer under live traffic.

'see "The Lottery" on hyperscale.rs

THE BOTTOM LINE

STANDOUT CHANGE

Cross-shard locks now provably resolve by deadline readings and abort deadlocked transactions, closing a whole class of stuck-state failures.

WHAT'S NEXT

Likely next: continuing the fee-hold and hold-cycle correctness work in the VM, and finishing the co-hosted validator seating path.

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

Recovery anchors made solid

Joining nodes re-read their anchor only when no peer holds it, and boundary pins are now retained in the production stores, not just the simulator. A restarted shard rebuilds from provable history.

'see "The Library" on hyperscale.rs

ALSO THIS WEEK

- Signatures are now verified one by one instead of trusting a batch total, closing a subtle forgery gap.
- Beacon candidates that arrive before their parent block are held until the parent exists.
- Provisions are refused for transactions a node can't route, avoiding wrong-shard deliveries.
- Simulator dispatch now draws runs from half the pool's bound upward, making tests faster.
- Orphaned documentation was reattached to the items it describes.

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: A big week for design debate: crypto support in the new VM, migration from Babylon, and a long thread about archive nodes and who should store history — plus some fun banter about bad movies.

> *"It's like running a database which doesn't just store all the tables, it stores every query ever made. It makes sense for a couple of infra players to do so… but not everyone"*
>
> — FLIGHTOFTHEFOX, IN THE COMMUNITY TELEGRAM

FLIGHTOFTHEFOX EXPLAINED

VM crypto plans

The VM will cover all major hash families, native signature verification (ed25519, secp256k1, ML-DSA65, maybe secp256r1 for passkeys), and complete native proof verifiers like SUI's fastcrypto.

Migration from Babylon

There's no closed door: even worst-case, existing contract state can be read out of Babylon, re-encoded, and re-homed into the new tree — he could rewrite every contract himself and it would still be fine.

Why validators shouldn't hoard

Validators only need current state to validate the next block; storing every old receipt just accumulates cost until operators quit — that's why archive nodes will be separate.

Native upgradability

He prefers native contract upgrade paths over hand-rolled proxy contracts, which in a sharded system drag two components — possibly on different shards — into needless overhead.

THIS WEEK'S CONCEPT

The Generals — *The two generals put down their messengers.* Most of this week was about cross-shard locks, deadlines and tie-breaks — making sure two transactions fighting over the same state can never leave the network stuck or half-done.

hyperscale.rs/generals

JARGON, DECODED

Hold cycle — When two transactions each hold a lock the other needs, so neither can finish — the network now detects and breaks these by a fixed rule.

Archive node — A special node that keeps far more history than validators need, so gateways and explorers can look up old data.

Deadlock — The stuck state where two parts of the system each wait forever for the other to go first.

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



98% Rust · 1% Config · 1% Docs

WHERE THE WORK LANDED

node · The Generals
lock and deadline rules tightened

types · The System
core data types reshaped

vm/kernel · The Journey
fee holds and credits enforced

scenarios · The Crash Lab
new multi-shard test scenarios

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 10 · since August 2026. Since day one: 5,326 commits — one developer, fully in public.

LINKS

WEBSITE
hyperscale.rs

DISCUSS THIS ISSUE ON X
x.com/i/status/2109509672247914868

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.