XI'AN · HYPERSCALE FOR RADIX

Hyperscale Weekly

Split the work, not the world

WEEK #17
31 Aug – 6 Sept 2026
Milestone 2 · Radix Engine
Week 5 · ~month 1 of 5

MILESTONE 2 PROGRESS 23%

Previously: flightofthefox hardened the VM to reject invalid constructions at build time, formalized vaults and proofs, and was mid-flight on the proof-and-authorization presenting paths.

This week the Radix network suffered a major VM-level exploit, sending shockwaves through the community. flightofthefox stayed focused on the code, building the network's first real garbage-collection system and tightening transaction signing — while being transparent about protecting Hyperscale's runway and sketching how a future migration would work.

19	+4.1k −711	93	0
-95% vs last wk	-80% vs last wk		
COMMITS	LINES CHANGED	FILES TOUCHED	DAY COMMIT STREAK

COMMITS / DAY  peak 18/day · 2/7 days active

WHAT FLIGHTOFTHEFOX BUILT

Unreachable state, swept clean

flightofthefox built a garbage-collection system: nullifiers track state that nothing can reach, grouped by expiry buckets so each block retires exactly what's due. The network can now reclaim storage automatically.

'see "The Library" on hyperscale.rs

Intents declare their network

Transaction intents now carry signed headers naming which network they're for and how long they're valid — stopping cross-network replay and tightening each intent's time window to the narrowest bound.

'see "The Journey" on hyperscale.rs

THE BOTTOM LINE

STANDOUT CHANGE

The substate sweep gives the network its first real garbage collection — automatically tracking and removing state that nothing can reach anymore.

WHAT'S NEXT

Likely next: finishing the intent-signing and network-validation paths, and integrating the substate sweep into the shard's execution lifecycle.

From the community

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

HEARD IN THE CHAT

The mood: The Radix network hack dominated all conversation, with debate over relaunching vs. migrating and flightofthefox transparently liquidating XRD to protect Hyperscale's runway.

> *"hard to hyperscale when you're homeress"*
>
> — FLIGHTOFTHEFOX, IN THE COMMUNITY TELEGRAM

FLIGHTOFTHEFOX EXPLAINED

Sharding localizes failures
One shard halting doesn't impact others, so liveness impact is localized. Prolonged outages get noted by the beacon chain, which rotates the whole committee if needed.

Migration to Xi'an
You'd dump Babylon state at a predetermined epoch, transform it, and load it at Xi'an genesis — same as Olympia to Babylon. The tricky bit is contracts need recompilation.

VM design comes first
The absolute number one priority is building an optimal VM perfectly suited to Hyperscale and sharding. Primitives kept from Scrypto are there because the ideas were genuinely good.

Formal verification limits
You verify smaller models in specialized languages, not the full code. It's not a silver bullet, but having invariants written down makes bugs far less likely.

THIS WEEK'S CONCEPT

The Library — *The moving day that never happens.* The bulk of this week built the state tree's cleanup machinery — tracking, indexing, and removing state that nothing can reach anymore.

hyperscale.rs/library

JARGON, DECODED

Nullifier — A marker the system uses to track state that should eventually be removed once it's no longer reachable.

Substate sweep — The network's automatic garbage collection: finding and removing state entries that nothing can reach anymore.

Expiry bucket — A way of grouping state items by the block height at which they should be cleaned up, so each block processes them efficiently.

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

95% Rust · 2% Docs · 2% Specs · 1% Config

WHERE THE WORK LANDED

storage-rocksdb · The Library
substate sweep and nullifier storage

vm/kernel · The Journey
intent signing and network validation

types · The Library
nullifier and sweep data structures

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 5 · ~month 1 of 5 · 23%. Since day one: 4,023 commits — one developer, fully in public.

LINKS

WEBSITE
hyperscale.rs

DISCUSS THIS ISSUE ON X
x.com/i/status/2096826179135164682

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.