HYPERSCALE-RS · THE BUILDER'S LOG

WEEK #26
2–8 Nov 2026

Hyperscale Weekly

Split the work, not the world

THIS WEEK'S WORK

Hardening the new engine's cross-shard reads so contracts can't see stale state

The new VM 100%

Previously: flightofthefox had just rebuilt the network layer around iroh and given staking pools self-founding and handover paths.

A quiet, focused week: all sixteen changes landed in the new VM, the engine built from scratch that lets smart contracts run across shards. flightofthefox spent the week tightening how the engine reads shared state and how fees are burned, while the chat was dominated by the fallout of a big exchange hack.

16	+1.9k −384	18	2
-83% vs last wk	-84% vs last wk		
COMMITS	LINES CHANGED	FILES TOUCHED	DAY COMMIT STREAK

COMMITS / DAY  peak 7/day · 3/7 days active

WHAT FLIGHTOFTHEFOX BUILT

Keeping shared state honest

Most of the week went into rules for how a contract reads state handed over from another shard: every read must come from the exact piece that was served, never from somewhere stale. This is what makes identical execution — and thus atomic cross-shard payments — possible.

'see "The Generals" on hyperscale.rs

Fees are burned, provably

The engine now tracks the fee burn alongside every state change, so destroyed fees are accounted for as rigorously as balances themselves. It matters because money vanishing without a record is exactly what an auditable network can't afford.

'see "The Journey" on hyperscale.rs

Random draws respect the clock

New rules pin committee draws to randomness seeds from strictly earlier epochs, and fix the clock per transaction rather than per block. That keeps validator assignments unpredictable and fair — nobody can peek at seeds before choosing how to act.

'see "The Lottery" on hyperscale.rs

THE BOTTOM LINE

STANDOUT CHANGE

The new VM now enforces that contracts can only ever read the exact cross-shard state that was proven and served to them.

WHAT'S NEXT

Likely next: more VM hardening around served-state reads and fee accounting, plus the hinted optimized networking design.

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 the Binance hack and whether Radix validators should block the stolen wallets; flightofthefox also held a lively Friday office-hours call.

> *"censorship resistance isn't something you can just assume, it has to be worked on"*
>
> — FLIGHTOFTHEFOX, IN THE COMMUNITY TELEGRAM

FLIGHTOFTHEFOX EXPLAINED

Validators are sovereign

He explained there's no coordinated vote needed to block certain transactions — validators can individually decide what blocks to vote on, and he reminded them they are sovereign.

Governance baked into protocol

He said Hyperscale has its own governance and voting built into the protocol, because any DAO-style voting not aligned 1:1 with actual consensus powers is toothless.

Post-quantum is mandatory

Asked whether Hyperscale will be quantum-resistant, he answered that it's not worth releasing anything non-quantum.

Friday office hours

He set up a drop-in TGIF voice call open to anyone wanting to chat about Hyperscale, Radix, crypto, software or life in general.

THIS WEEK'S CONCEPT

The Generals — *The two generals put down their messengers.* The bulk of the work hardens the engine's reads of provisioned state, which is the bedrock of executing cross-shard transactions byte-identically on every shard.

hyperscale.rs/generals

JARGON, DECODED

Fee burn — Transaction fees that are permanently destroyed rather than paid to anyone.

Kernel — The VM's trusted core that contracts must go through to read or write any state.

Provisioned state — A piece of shard state shipped to another shard under a proof, so it can be verified rather than trusted.

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



89% Rust 11% Docs

WHERE THE WORK LANDED

vm/kernel · The Generals
rules for reading served, proven state

vm/types · The Journey
fee burn carried with state changes

vm/docs
documenting the new VM rules

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 FOUR PARTS

done	100%	–	–	–
Adaptive Sharding done Aug 2026	The new VM	Indexing & APIs	Validator tooling & testnet	Foundations & reliability

Share of lines changed this week, by part; shared groundwork counts as Foundations & reliability. Since day one: 5,733 commits — one developer, fully in public.

LINKS

WEBSITE

hyperscale.rs

DISCUSS THIS ISSUE ON X

x.com/i/status/2119656495709057320

CODE (PUBLIC)

github.com/hyperscalers/hyperscale-rs

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.