

Cross-provider block reconciliation for Solana Yellowstone consumers

Satyawan Singh

25 October 2026

Repository	https://github.com/ss1738/yellowstone-reconcile (Apache-2.0, CI passing)
Project page	https://reconcile.satyawansingh.com
Category	Developer Tooling
Request	USD 15,000
On-chain accounts	None. Off-chain consumer tooling; no program deployed.

The gap, in the Foundation’s own words

Solana’s Alpenglow integration guidance directs stream consumers to buffer per connection, then stops at the layer above:

Run one buffer per connection. `bank_id` means nothing outside the stream it arrived on.
[...] Reconciling across connections is still yours to write.

The same page warns that merging on (`slot`, `bank_id`) across streams **fails silently** and returns “blocks fused from unrelated banks”, and states that `yellowstone-block-machine` does not cover cross-connection reconciliation.

Alpenglow activated on mainnet in epoch 1064 on 21 October 2026. The Foundation marks it a breaking change that requires indexing changes. Any consumer reading from more than one provider is exposed to this now rather than at some future date.

What has been built, unpaid, before applying

A reusable Rust library that takes one upstream `BlockStream` per connection and never mixes their events. It reconciles sealed blocks on blockhash, parent and counts rather than on `bank_id`, requires a configurable number of distinct providers before emitting, quarantines disagreement, reports conflicts explicitly instead of selecting silently, keeps provider identity stable across reconnects so a reconnect is not a second vote, and never fuses payload fragments from different providers.

Six properties are measured and reproducible from the repository, not asserted:

Property	Measured	Source
Silent provider tolerated before the bound rejects	27.1 s, 128 blocks at 212 ms/slot	<code>stall_budget</code>
Retention before a slot is refused as stale	868 s, 4,096 slots	<code>stall_budget</code>
Payload held at the bound, real mainnet block sizes	570.8 MiB	<code>payload_variance</code>
Same, were every block the largest sampled	1,011 MiB	<code>payload_variance</code>
Restart re-emitting a committed slot	never, across 3 tests	<code>tests/restart.rs</code>
Sink writes, crash plus full replay of 20 slots	20 rows, 20 writes, none lost	<code>durable_sink</code>

Example names above are files under `examples/`.

The slot rate of 212 ms was measured on mainnet-beta on 25 October 2026 over 1,000 and 5,000 slot windows. Mainnet block sizes were sampled the same day: median 4.26 MiB, largest 7.90 MiB, smallest 2,389 bytes. An earlier assumption of a round 2 MiB understated the memory ceiling by roughly half, and that figure was corrected in place rather than replaced quietly.

Also delivered: a wire guard that rejects malformed blockhashes and parents at this library's own boundary rather than panicking, an offline replay CLI, adversarial fixtures, and independent reproduction of both known defects in the upstream crate, recorded in `docs/DEPENDENCY-REVIEW.md`.

None of the above is billed in this request.

What the funding covers

Milestone	Deliverable and acceptance	Hours	Request
1	Agree the integration and recovery contract with a pilot integrator; integrate against a real durable store, replacing the in-process sink demonstration	45	USD 4,050
2	Validate two independent provider implementations; publish sanitised captured fixtures where permitted, representative payload memory and throughput, sustained soak results and unresolved gaps	70	USD 6,300
3	Resolve independent review findings; publish migration and example integration, maintenance policy and a first supported release	35	USD 3,150
	Infrastructure: paid subscriptions to two independent Yellowstone providers		USD 1,500
	Total	150	USD 15,000

The rate assumption is USD 90 per hour for Rust and distributed-systems correctness work. It is an assumption, not an agreed rate or an external quote. The infrastructure line has no supporting provider quotation and is the least certain figure here. Purchasing a formal security audit is not included; remediating an independent review is.

An earlier draft of this budget requested 280 hours. The reduction to 150 reflects work since completed and not billed, and a rate that reflects the work rather than one chosen to make the total look modest.

Success measures

Measured rather than asserted, with every baseline currently zero:

- No cross-connection raw-event mixing in the published contract or the regression suite.
- Reconnect and replay produce no duplicate sink effects in the documented durable integration.
- Conflicting qualified block reports produce explicit failures, never silent selection.
- Bounded buffering is measured against representative payload sizes with an unavailable peer.
- A named integrator reproduces the example and gives written acceptance of its usefulness.

The last of these is a proposed acceptance criterion, not an existing adopter.

Risks, stated plainly

No adopter, no pilot, no upstream endorsement. Milestone 2 is the least defined for exactly that reason: “two independent providers” names no provider yet. A scoping question asking whether this layer belongs upstream or as a companion crate is open at `rpcpool/yellowstone-block-machine#23` and unanswered at the time of writing.

Dependency risk. The upstream dependency is currently pinned to the head of an open, unmerged pull request rather than to its default branch, because the defect it fixes is pruned inside upstream `BlockStream` and cannot be corrected downstream. No external pull request has yet been merged in that repository.

Scope may belong upstream. If the maintainers would rather absorb this as a module or an example, the right outcome is to contribute there and not to maintain a parallel implementation. That would reduce this request rather than redirect it.

Agreement is not validity. Provider agreement cannot establish ledger validity or provider independence, and waiting for multiple qualified providers trades availability for consistency. Both are documented rather than hidden.

All current measurement is synthetic. No production incident, incidence rate, adoption count or performance advantage is claimed anywhere in this proposal. The library has not been audited.

Applicant

Satyawan Singh, applying as an individual. Background is AMM and protocol internals rather than Solana: Uniswap v4 hook and fixed-point math, with an open pull request against `solana-program/token-2022` fixing an unresolved arithmetic audit finding. This is a first Solana library. The work above was completed and published before applying.