

Litepaper

The open quoting seat

A prop AMM on Solana where bonded quoters price a shared SOL/USDC vault on their own books, inside limits the program enforces on every fill.

Version 1.0 October 2026 Program wire version 5

Contents

The open quoting seat

- Where prop AMMs stand
 - What OpenProp changes
 - The shape of the system
-

Quotes and fills

- A book is a ladder
 - The five checks
 - The reference price
 - Walking the ladder
-

Measuring and settling a book

- Excess over a passive hold
 - The high-water mark
 - The split
 - \$OPENPROP
 - Losses fall on the bond first
 - Epochs and checkpoints
 - The LP side
-

Allocation follows performance

- The score
 - Tiers set the leverage
 - Sharing out the vault
 - Entering and leaving the seat
-

Limits the program enforces

- Book limits
 - Pair limits
 - Parameter changes
 - The guardian
 - Incidents are records
-

The service

- Why Solana
- What runs where
- Using it

v0 scope

The open quoting seat

OpenProp is a proprietary AMM on Solana whose quoting seat is open. A shared SOL/USDC vault holds the liquidity. Bonded quoters each run a book on a slice of that vault, priced by their own models, and every fill is checked by the program before it happens. Takers reach the books through the aggregators they already use; liquidity providers deposit once and are paid from what the quoters earn above a passive hold.

Where prop AMMs stand

On Solana, most SOL/USDC volume now clears against proprietary AMMs: venues where one market maker updates its own quotes on chain every slot and aggregators route swaps into them. They price tightly because the maker's model, not a curve, sets the price, and because Solana makes a quote update cheap enough to send continuously.

Each of these venues is closed. One firm writes the quotes, provides the capital and keeps the result. A maker who could price the pair well has no way to quote there without building and funding a venue of its own, and a liquidity provider has no way to put capital behind the makers who price best.

What OpenProp changes

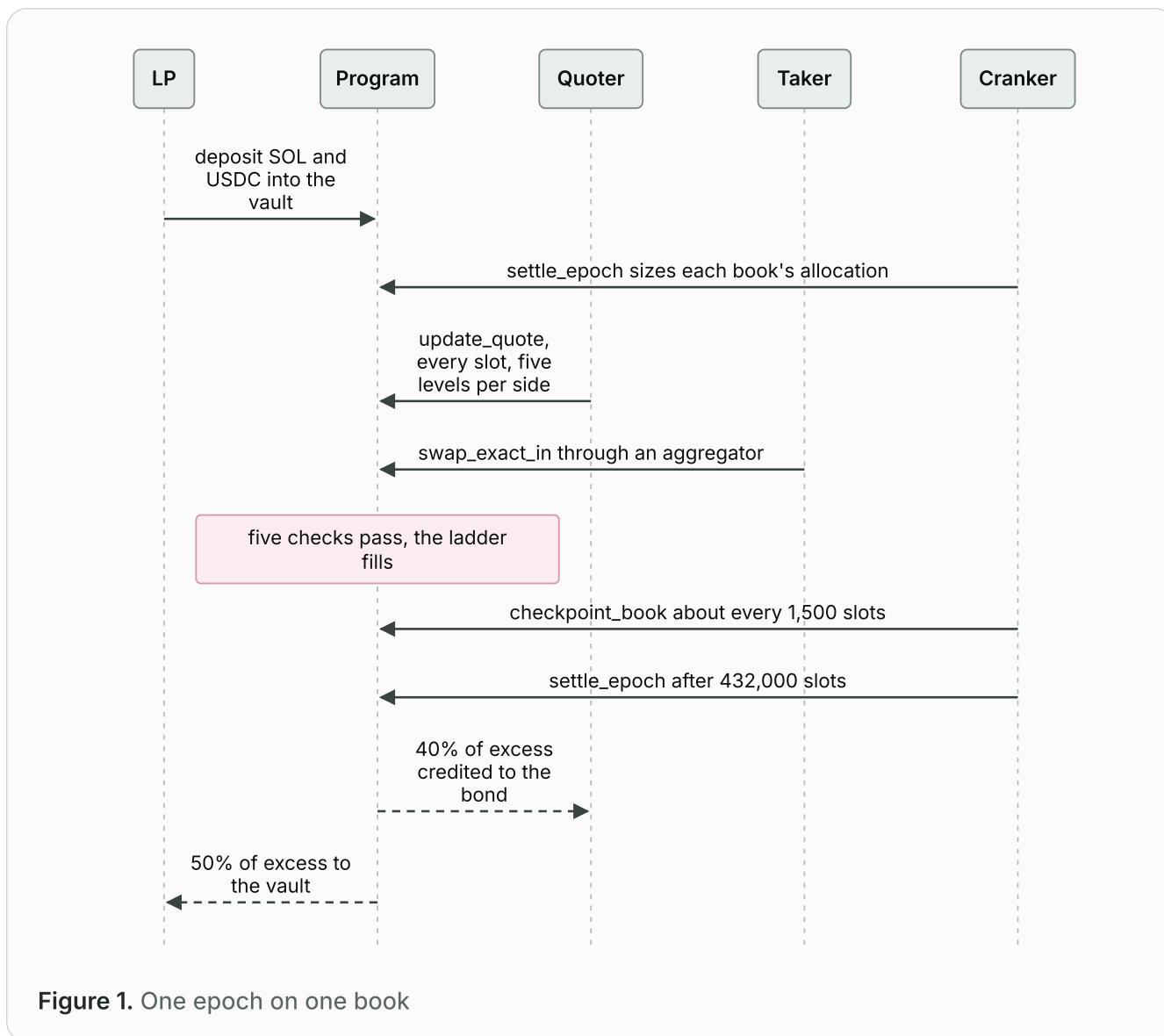
OpenProp separates the three roles a prop AMM fuses together:

Role	Who	What they bring	What they receive
Liquidity provider	Anyone holding SOL and USDC	Capital, into one shared vault	50% of positive excess above each book's high-water mark, plus the vault's hold return
Quoter	Bonded market makers, quants, agent teams	A pricing model and a USDC bond	40% of positive excess, credited to the bond
Taker	Swaps from wallets and aggregator routes	Order flow	Fills at the quoted price inside the program's checks

The protocol's remaining 10% buys \$OPENPROP, the protocol's token, on the market and burns it.

The shape of the system

A single program owns the vault, the books and the quotes. A quoter registers with a bond, receives an allocation of the vault, and posts a ladder of five price levels per side with a compact `update_quote` instruction. A taker's `swap_exact_in` walks the best book's ladder after five checks pass. Cranker services run checkpoints during the epoch and settle each book when the epoch ends, moving the excess into the right hands and resizing allocations for the next one.



The rest of this paper walks through each step: how a quote is posted and filled, how a book is measured against a passive hold and settled, how allocation follows performance, and how the program stops a book or the pair the moment its limits are crossed.

Quotes and fills

A book is a ladder

Each quoter's book holds a ladder of up to five levels on each side of a mid price. A level is a price offset from the mid and a size in lamports of SOL. The quoter's model decides the mid, the offsets and the sizes; the program decides whether the book may trade at all.

Quotes are written with `update_quote`, the hottest instruction in the program. It carries the mid, the level offsets and sizes in a fixed little-endian layout, and touches one account, the book's `Quote`. Its compute target is 200 CU; the measured cost is 178 CU. At Solana's 200 ms slot time, a quoter can rewrite its whole ladder every slot for a small fraction of a cent.

A quoter signs updates with a hot authority — up to three hot keys per quoter — while the bond and the book's limits stay under the owner key. The hot key can only move quotes; it cannot withdraw anything.

The five checks

Every `swap_exact_in` names a book. Before any level fills, the program runs five checks against that book and the pair:

Check	Passes when
Fresh quote	The quote was written no more than 2 slots ago
Inside band	The quoted mid sits inside the band around the reference price
Skew limit	The book's inventory skew after the fill stays within its limit
Loss window	The book is inside its hourly limit (0.5% of allocation) and its epoch limit (2%)
Breaker	The pair's circuit breaker is not tripped

If a check fails, the swap does not fill on that book, and the transaction leaves every account as it was. OpenProp's adapter for Jupiter's AMM interface runs the same rule crate as the program, so when it quotes a book that would fail a check it returns no route.

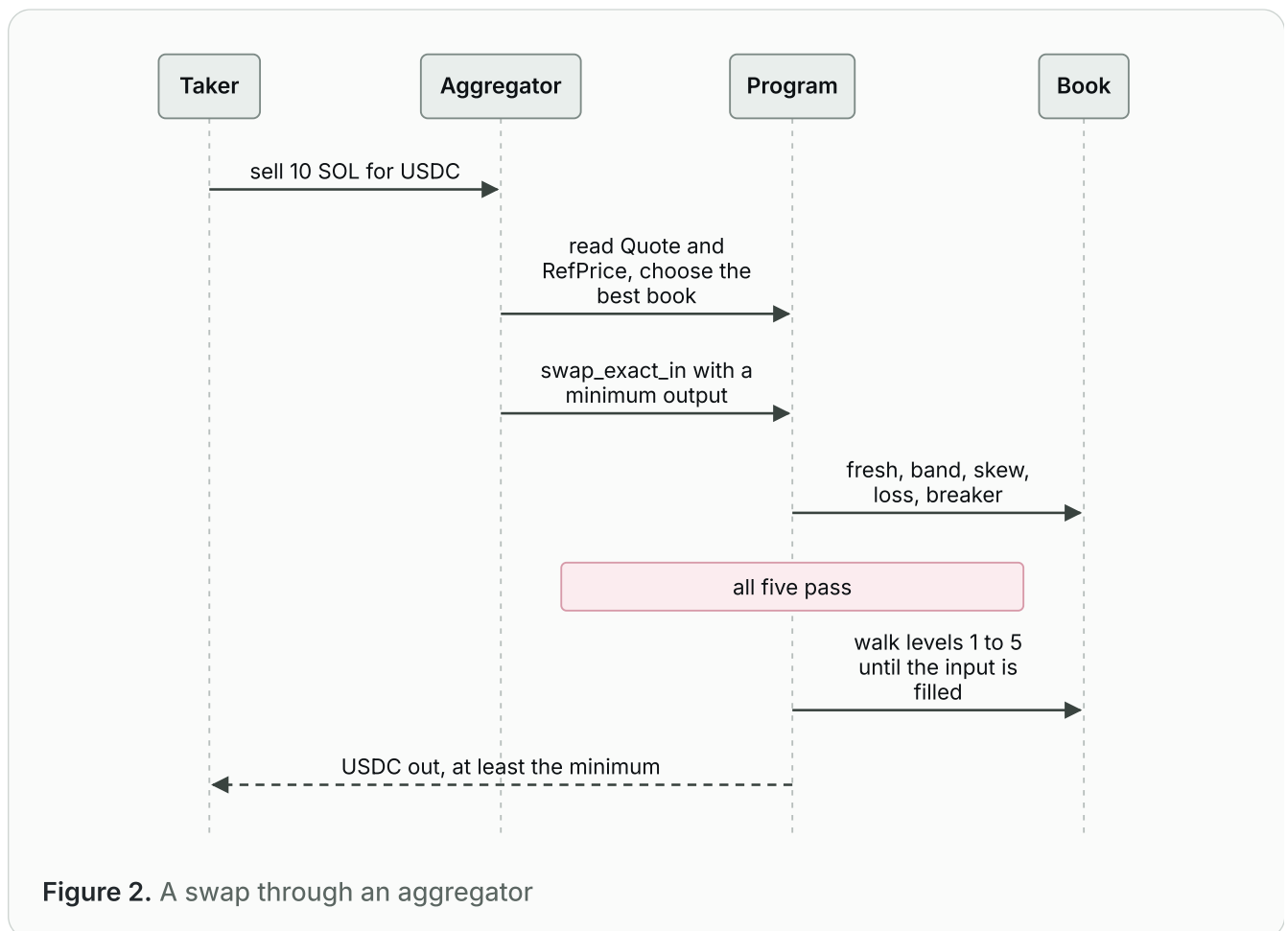
The reference price

The band, the breaker and every checkpoint measure against a reference price kept in the `RefPrice` account. It is the integer midpoint of two independent oracle sources, Pyth Pro and Switchboard Surge, posted with each source's signed payload and verified on chain against the `Ed25519` precompile and the `SlotHashes` sysvar. Its confidence is the wider of Pyth's own interval and half the

gap between the two sources. A reference older than 10 slots, or one whose confidence is too wide, halts the pair rather than letting a book trade against a stale anchor.

Walking the ladder

A fill walks the book's levels from the best price outward, taking as much of each level as the input allows, until the input is spent or the ladder's depth or skew boundary is reached. The output is computed in integer arithmetic by the program. The taker's transaction states the least it will accept, and a fill that would deliver less fails as a whole with `SLIPPAGE_EXCEEDED`.



A swap that walks all five levels measures about 20,900 CU. The most expensive measured swap, a partial fill at the skew limit on a large inventory, costs 37,497 CU, inside the 40,000 CU estimate that aggregator routes budget for.

Measuring and settling a book

Excess over a passive hold

A book is judged against the simplest alternative: holding the inventory it was given. At the start of an epoch the book receives an allocation of SOL and USDC from the vault. At every checkpoint the program marks two values at the reference price P :

$$V_{\text{book}} = s \cdot P + u \quad V_{\text{hold}} = s_0 \cdot P + u_0$$

where s, u are the book's tracked SOL and USDC now and s_0, u_0 the amounts it started the epoch with. The **excess** is the difference:

$$X = V_{\text{book}} - V_{\text{hold}}$$

A quoter who buys low and sells high around the reference earns positive excess; one who is picked off by faster flow loses it. Price moves that would have happened to a holder anyway cancel out, so the measure isolates what the quoting added. The tracked inventory changes only through swaps and settlement, so tokens sent to a book from outside never count as performance.

The high-water mark

Each book keeps a high-water mark of its cumulative excess. Only excess above the mark is shared; a book that loses and then recovers pays nothing on the recovery, because that ground was already paid for.

The split

When the epoch ends, `settle_epoch` moves each book's positive new excess above its mark three ways:

Recipient	Share	Where it goes
Liquidity providers	50%	Stays in the vault, raising the value of every share
Quoter	40%	Credited to the quoter's USDC bond
Protocol	10%	Buys \$OPENPROP on the market and burns it

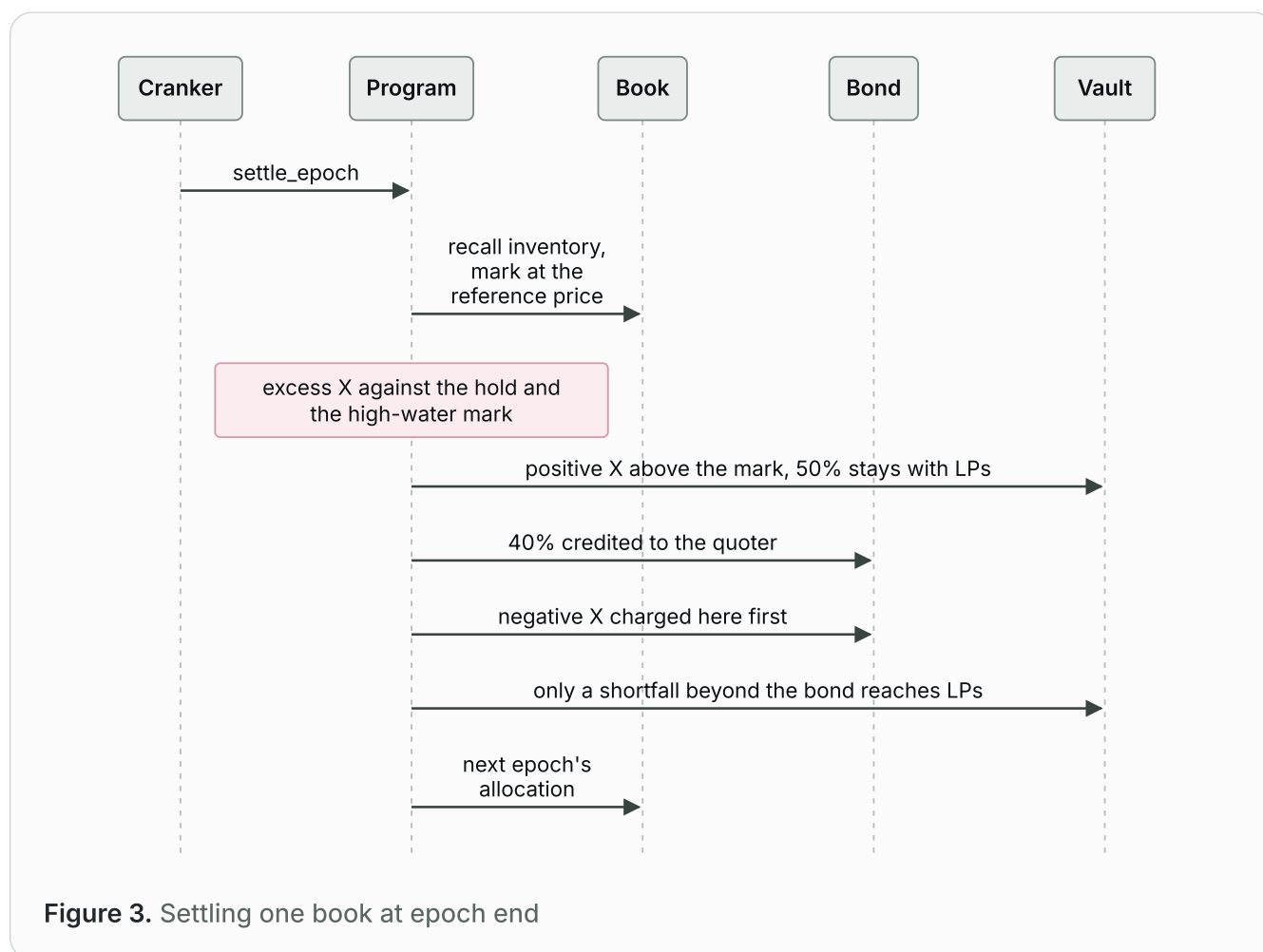
\$OPENPROP

\$OPENPROP is the protocol's token: 1,000,000,000 fixed supply, all of it sold in one public launch on equal terms, with no presale, no team, investor or airdrop allocation and no vesting. The

protocol's 10% is paid in USDC at each settlement into an account only the program controls. Its one way out is a single transaction that swaps it for \$OPENPROP through Jupiter and burns every token bought; the program fails the whole transaction unless the supply falls by at least the quoted amount. The burn follows excess, not volume: a fill earns the protocol nothing on its own, and only what the books earn above their high-water marks is shared, so a venue whose quoter's price well burns more.

Losses fall on the bond first

When a book's excess for the epoch is negative, the loss is covered in a fixed order: first the quoter's tracked USDC bond, then any of the quoter's profit credit not yet paid out, and only after both are spent does it reach the vault. The bond is first-loss capital in the literal sense, and coverage drawn from it restores the book's cumulative excess toward its mark.



Epochs and checkpoints

An epoch is 432,000 slots, about 24 hours at a 200 ms slot. Inside it, a checkpoint runs about every 1,500 slots — roughly every five minutes — recording the book's excess return for scoring and updating its hourly and epoch loss windows. Checkpoint events are written on chain in a fixed 210-

byte layout and settlement events in a 242-byte one, so every number in the app and the API is rebuilt from events anyone can read.

The LP side

A liquidity provider deposits SOL and USDC and receives vault shares. Deposits take the largest basket that matches the vault's live ratio, up to the amounts the depositor states, and leave the rest in the wallet. Shares are minted against the vault's net asset value; a `minShares` floor makes the deposit fail rather than issue fewer than expected.

To leave, a provider requests a withdrawal. The shares move to pending, the next settlement reserves their proportional basket of SOL and USDC, and the basket becomes claimable one epoch after the request — two epochs at most. The wait lets every book be recalled and settled before assets leave, so a withdrawal always takes its share at a settled price.

Allocation follows performance

The score

At each checkpoint the program records the book's excess return r_t — its excess since the last checkpoint divided by its allocation. The score is a shrunk, exponentially weighted ratio of mean to volatility:

$$S = \frac{\bar{r}}{\sigma_r} \cdot \frac{n}{n + n_0}$$

$\bar{r}$ and σ_r are an EWMA mean and standard deviation of the checkpoint returns with a 7-day half-life, so recent quoting counts more than old quoting. The factor $n/(n + n_0)$ pulls a short record toward zero: a new quoter cannot win a large allocation from a handful of lucky checkpoints. All of it is computed on chain in integer arithmetic, with an integer square root for σ_r .

Swaps signed by the quoter's own keys — its owner and its hot keys — stay in its profit and loss but are left out of the score.

Tiers set the leverage

A quoter's allocation is bounded by its bond times its tier's leverage:

Tier	Allocation up to	To reach it
0	1× bond	Register with a USDC bond
1	5× bond	At least 14 epochs, a positive score, no loss-limit trips in the last 7 epochs
2	15× bond	At least 45 epochs, a score above the tier-2 threshold, a guardian review in v0

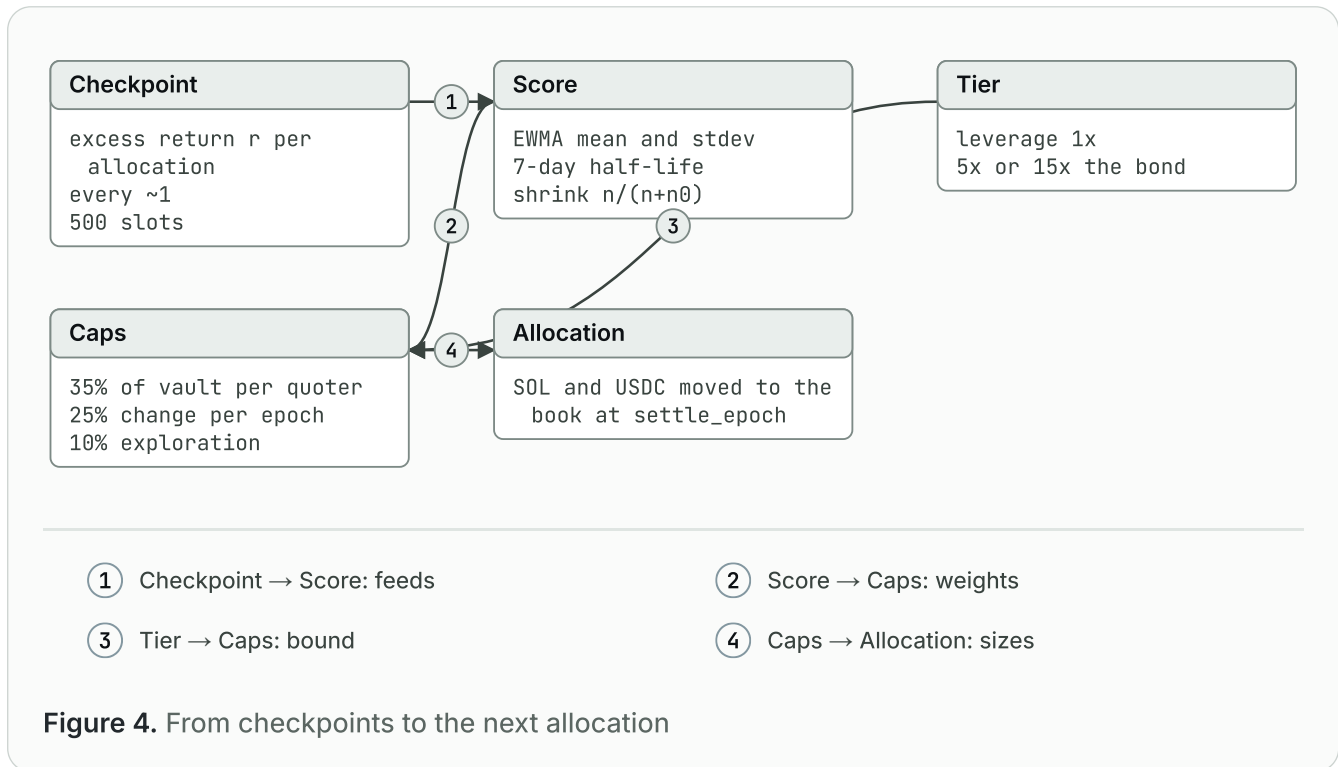
Two epoch-loss trips within 7 epochs drop a quoter one tier.

Sharing out the vault

At each settlement the program sizes the next epoch's allocations from the scores. Books with a positive score share the deployable vault in proportion to their score, under three caps:

- **Concentration.** No quoter holds more than 35% of the vault.
- **Change.** A book's allocation moves by at most 25% per epoch, so capital shifts toward better quoters steadily instead of all at once.
- **Exploration.** 10% of the deployable allocation is set aside and split across tier-0 books, so a new quoter can build the record the score needs.

A paused or halted book receives no new allocation. Its inventory returns to the vault at settlement.



Entering and leaving the seat

Registration takes a USDC bond and creates the quoter's book and quote accounts. The vault in v0 has room for five registered books on SOL/USDC; a seat frees when a fully unbonded, flat quoter deregisters.

A quoter adds bond at any time. Taking bond out starts an unbond request that waits at least two epochs, and only the part not backing a live allocation or a profit lock can leave. Profit credited to the bond matures in tranches before it becomes withdrawable.

Limits the program enforces

Every limit in OpenProp is checked by the program itself, on the transaction that would cross it. Off-chain services report and resume; they do not stand between a taker and a fill.

Book limits

Limit	Value	What happens
Quote age	2 slots	The book does not fill until it quotes again
Hourly loss	0.5% of allocation	The book stops filling until the window rolls
Epoch loss	2% of allocation	The book stops filling for the rest of the epoch
Repeated trips	2 epoch-loss trips in 7 epochs	The quoter drops one tier

Loss is projected before the fill: a swap that would push a book past its limit is rejected, not filled and then flagged. The quoter can also pause its own book at any time with its owner key; a paused book keeps its bond and history and resumes when the owner says so.

Pair limits

The circuit breaker covers every book at once:

Condition	Threshold
Reference too old	More than 10 slots
Reference too uncertain	Confidence wider than the configured gate
Sources disagree	Pyth Pro and Switchboard Surge further apart than the divergence limit
Fast move	More than 3% within 150 slots

When any condition holds, swaps halt on the pair. Fills resume 50 slots after the condition clears, so a recovering market is quoted against a reference that has settled. Checkpoints and settlement keep running through a halt, valuing books at fresh reference prices while no one trades.

Parameter changes

Parameters live in the Config account and change through an admin multisig. Tightening a limit applies immediately. Loosening one waits 3 epochs, and never less than the longest withdrawal wait, so every liquidity provider can leave under the rules they deposited under before a looser rule

takes effect. No instruction lets the admin, the guardian or anyone else move vault or book assets outside the deposit, swap, settlement, withdrawal and bond paths.

The guardian

A guardian key can halt the pair and attests the tier-2 review. A separate halt-only key can be given to an automated monitor: it can stop fills and nothing else. Resuming after a manual halt is an admin action and goes through the same 50-slot recovery as an oracle halt.

Incidents are records

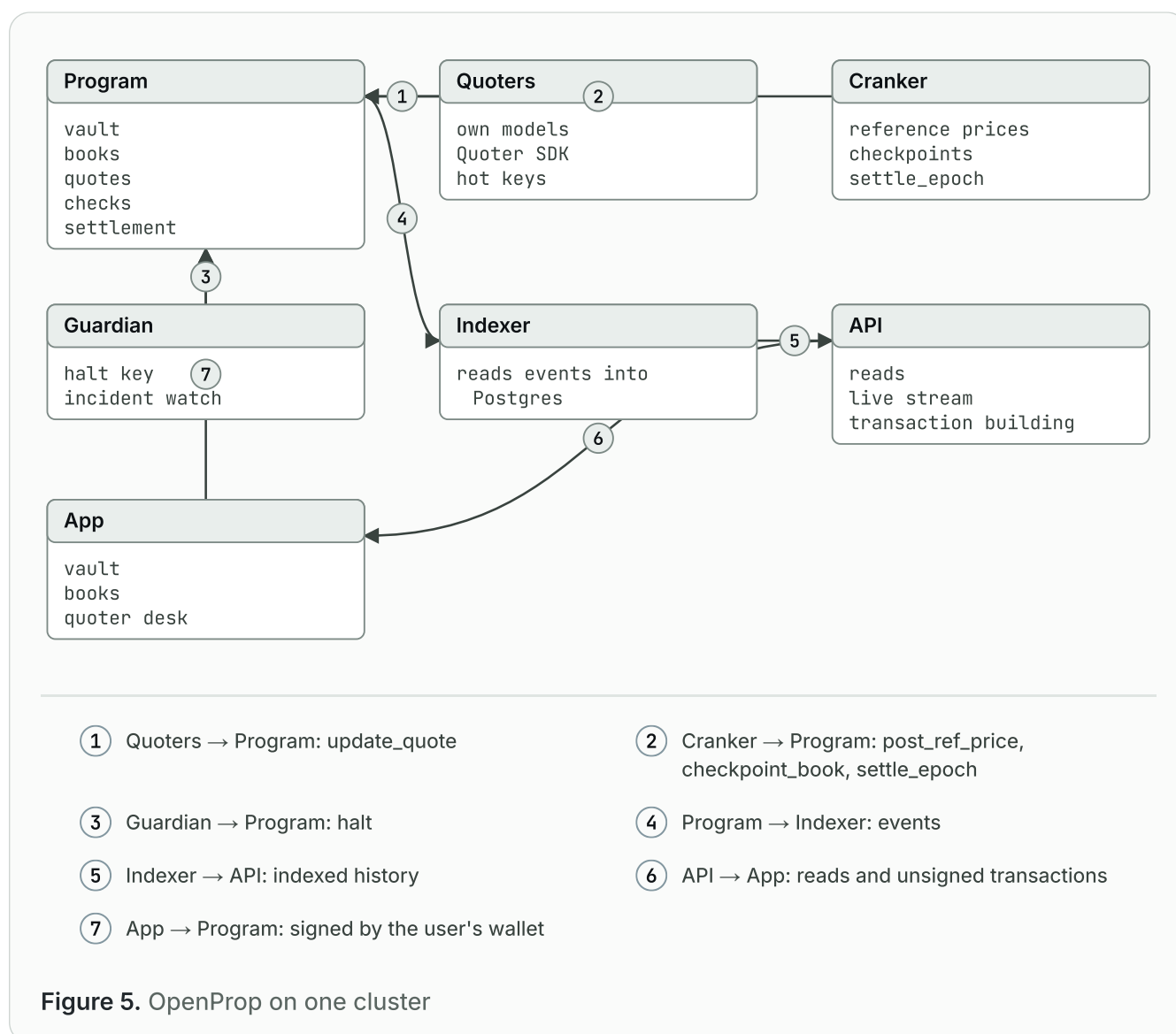
Every trip — a stale book, a loss limit, a breaker halt, a manual halt — is written on chain as an incident event. The app's Incidents page and the API's incident feed are built from those events, with the slot, the book and the measured value that crossed the line.

The service

Why Solana

A prop AMM lives on cheap, frequent quote updates and fast confirmation. Solana's 200 ms slots let a quoter rewrite its ladder five times a second. A 178 CU `update_quote` uses a sliver of the 1.4 million CU a transaction may spend, and the Quoter SDK sends each update down the landing paths a quoter plugs in, in order — BAM's Maker Priority Plugin, the Block Engine, the TPU — so a fresh quote can reach the leader ahead of the flow that would trade on a stale one.

What runs where



The program is the venue. The cranker posts reference prices from Pyth Pro and Switchboard Surge, runs checkpoints and settles each epoch; anyone may run those instructions, and the

program checks every input. The indexer turns on-chain events into the history the API serves. The API also builds unsigned transactions for the app, which the user signs in their own wallet. No server holds a key that can move user funds.

Using it

- **Liquidity providers** connect a Solana wallet in the app, deposit SOL and USDC into the vault, follow each book's excess and the epoch's settlements, and request a withdrawal when they want to leave.
- **Quoters** register with a USDC bond, delegate a hot key to their quoting process, and stream quotes with the Quoter SDK. The app's quoter desk shows the book's checks, loss windows, score and tier as the program sees them.
- **Takers** swap in the app against the best live book, or through any route that integrates the venue; OpenProp's adapter implements Jupiter's AMM interface and evaluates the same rule crate as the program.
- **Builders** read everything — books, quotes, checkpoints, settlements, incidents, parameters — from the public API and its live stream, documented in the Docs.

v0 scope

The first deployment runs one pair, SOL/USDC, with a vault cap of \$2,000,000 and room for five quoters. The protocol's share of excess is the only fee, and it buys and burns \$OPENPROP; there is no deposit fee and no withdrawal fee.