

Litepaper

Anyperp

Perpetual futures on an artist's monthly listeners, on Solana. Each side's open interest stops below the cost of moving the number, so faking it loses money.

Contents

Anyperp litepaper

The product

The problem every metric market has

Caps priced to manipulation

The two-root oracle

Next-print trading

The vault and its economics

Listing

Settlement

Programs and services

The ANYPERP token

Launch parameters

The product

Anyperp is a perpetual futures service on Solana for one kind of number: an artist's monthly listeners, the count on the artist's public profile page on a streaming platform. Each listed artist is a market. Its index is the listener count divided by a per-market divisor, printed every 15 minutes. A position goes long or short that index with USDC margin, isolated per position, at up to 3× leverage, and has no expiry. A trader who expects an audience to keep growing holds one position for as long as the view lasts. A trader with a view on two artists holds a long in one market and a short in the other.

One USDC vault takes the other side of every market. Liquidity providers deposit into it, receive shares, and earn 60% of trading, liquidation and borrow fees, plus net funding and the vault's side of trader PnL. The service launches with 20 artists, and anyone can propose another by posting a bond.

Every price is the median of two independent attestations, verified on chain, and every order fills at a price nobody has seen yet. Each side of each market is capped below the cost of moving the number, so manipulating it loses money. This paper describes those mechanisms, the vault behind them and the path a market takes from proposal to settlement.

The problem every metric market has

Prediction venues already list markets on internet numbers, monthly listener counts among them. They are dated contracts: each resolves on a single snapshot, liquidity resets every period, and a trend can be held only by rolling from one contract to the next. A perpetual on the level gives continuous exposure in one market per metric.

Any market on a metric has to answer two questions. The first is who attests the number. A web page has no oracle network behind it, and a market whose deployer sets the price, or whose price comes from one vendor, inherits that party as its trust base. The second is what stops someone from moving the number. Metrics can be bought: streaming fraud is an industry, and one scheme ran more than 1,000 bots and collected over \$8M in royalties. A perpetual whose open interest exceeds the cost of moving its index pays whoever moves it.

Anyperp answers the first question with two independent attestation roots whose signatures the program verifies, and the second with arithmetic. Each side's open interest stays below the cost of moving the metric far enough to profit, so an attack loses at every size.

Caps priced to manipulation

The derivation

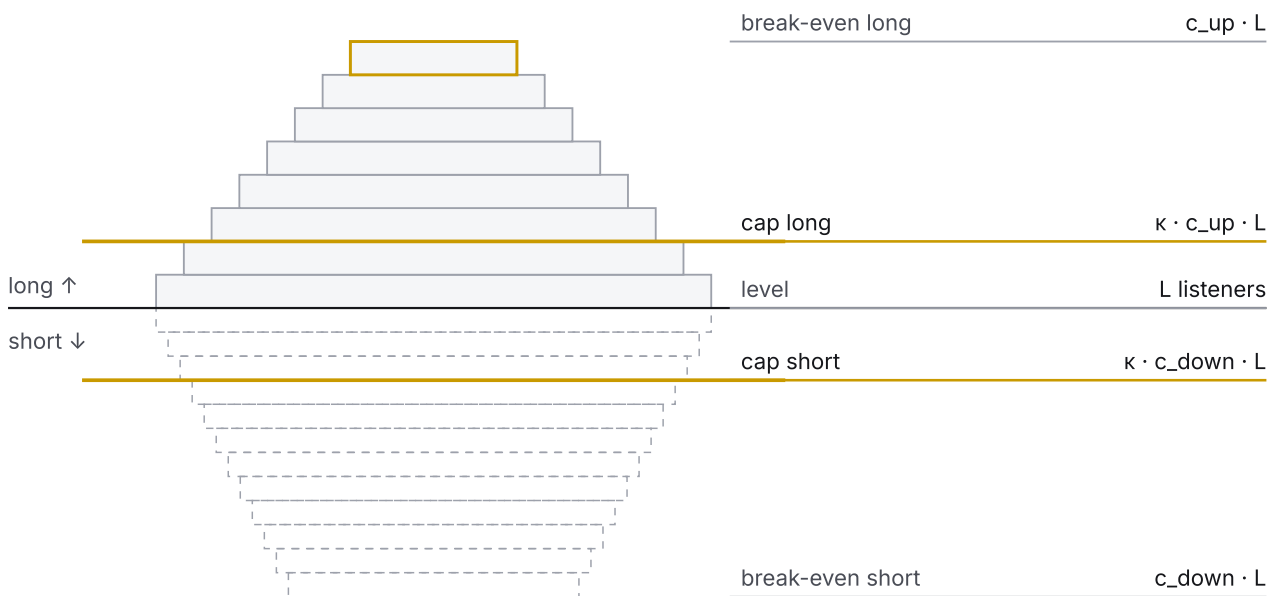
Take an attacker holding long notional N who can push the metric up by a fraction x . If the cost of the push is linear in the listeners added, $C(x) = c_{up} \cdot x \cdot L$, where L is the metric's level and c_{up} the net cost of one added listener held through the print, the attack's profit is

$$\pi(x) = Nx - c_{up} x L = x(N - c_{up}L)$$

For every $x > 0$ this is negative whenever $N < c_{up}L$. The break-even open interest is $c_{up}L$, and it doesn't depend on how far the attacker pushes. Anyperp caps each side at a fraction κ of its break-even:

$$\text{cap}_{long} = \kappa \cdot c_{up} \cdot L \quad \text{cap}_{short} = \kappa \cdot c_{down} \cdot L \quad \kappa = 0.25$$

With $\kappa = 0.25$, break-even sits four times above the cap.



A market as a terraced hill: its height is the break-even open interest $c_{up} \cdot L$, the κ contour sits at a quarter of it, and the basin for shorts is three times deeper.

What goes into the cap

- c_{up} is the cost of one fake unique listener that survives the platform's filtering of artificial streams.
- **The subject's cost.** An artist or label gains visibility from a higher count, so its net cost of inflating it is lower than an outsider's. Caps use that lower, subject's cost.

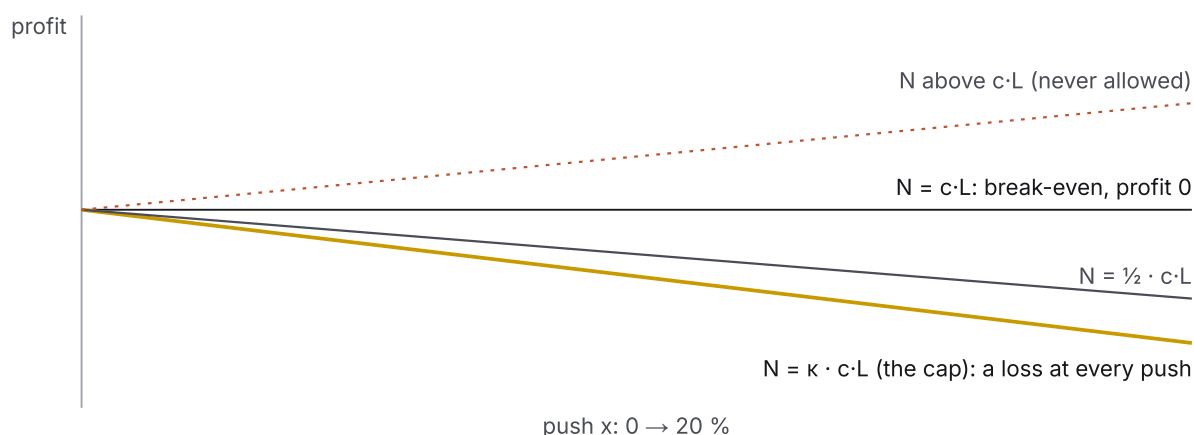
- $c_{down} > c_{up}$. Removing real listeners is much harder than adding fake ones, so the short cap is set from its own, higher cost and sits above the long cap.
- **Global per side.** A cap bounds the total open interest of a side across all accounts, so splitting a position across wallets doesn't raise it.
- **Scale.** Caps grow with L : a larger audience costs more to move and supports a larger market.

A worked example

For an artist at 60 million listeners with $c_{up} = \$0.02$ and $c_{down} = \$0.06$ per listener (example values), break-even long open interest is \$1,200,000 and the caps are:

	Break-even	Cap ($\kappa = 0.25$)	Probation cap (25% of the cap)
Long	\$1,200,000	\$300,000	\$75,000
Short	\$3,600,000	\$900,000	\$225,000

An attacker holding the whole long cap, \$300,000, who pushes the count up 10% gains \$30,000 on the position. Adding 6 million listeners costs \$120,000. The attack loses \$90,000. At four times the cap, \$1,200,000, the same push gains \$120,000 against \$120,000 of cost: break-even, before fees.



Attack profit against push size: below the cap it is negative for every push; at break-even open interest it is zero.

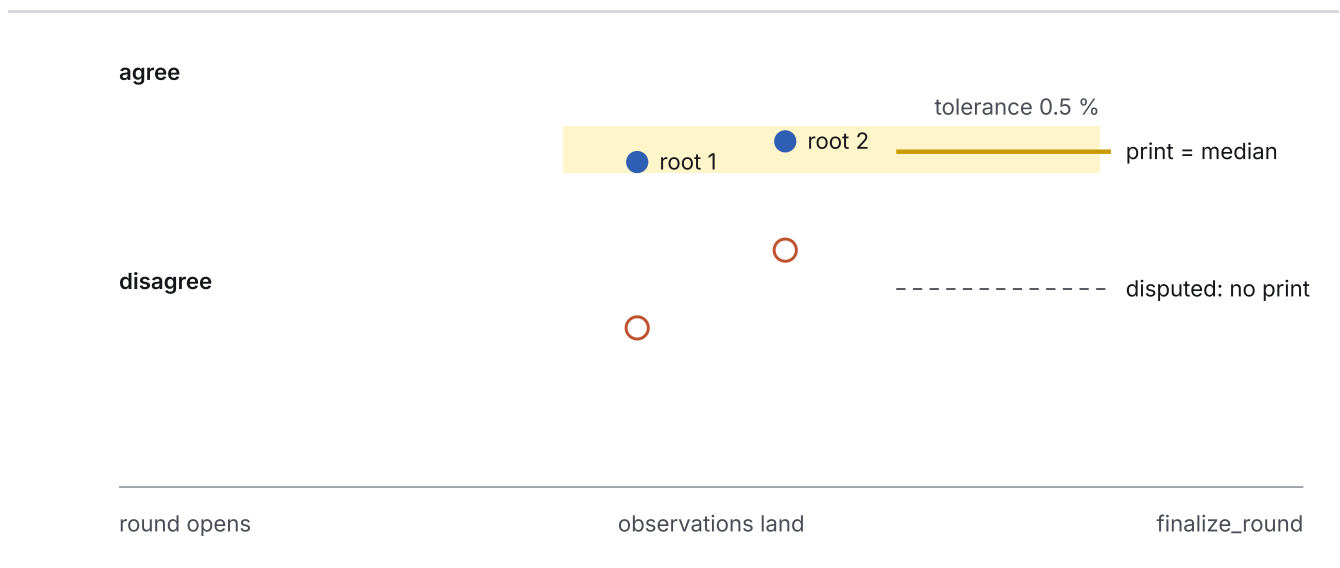
Two other mechanisms work alongside the caps. A print more than 10% away from its value 24 hours earlier doesn't take effect at once: the round is held and opens on the side of the move pause until later rounds confirm the level, so nobody can add exposure behind a sudden push. And because orders fill at the next print, a position has to be in place before the number moves; nobody can buy into a move they already see.

The two-root oracle

A round opens every 15 minutes for each market. Two attestation roots read the source in that round:

Root	Signer	Verified with
Root 1	a Reclaim attester, identified by its Ethereum-style address	Solana's secp256k1 precompile
Root 2	an independent notary run by a separate operator	Solana's Ed25519 precompile

Reporters fetch the count through each root and post each signed observation, root 1's with `submit_reclaim_observation` and root 2's with `submit_notary_observation`. Reporters are permissionless: validity comes from the root's signature, which the program checks in the same transaction against that root's allowlisted keys. The signed claim binds the program, the metric and the hash of its spec, the round and its sequence, the root, the value, the fetch time and the source time. Both times must fall inside the round's 15-minute window.



One round: it opens, both roots post observations, the tolerance check runs, and the round prints at the median or records a dispute.

`finalize_round` writes a print only when both observations are present, agree within 0.5% of the smaller value, and were fetched later than the previous print's. The print is their median divided by the market's index divisor:

$$I_t = \frac{v_1 + v_2}{2d} \quad |v_1 - v_2| \leq 0.005 \cdot \min(v_1, v_2)$$

When the roots disagree, the round is `disputed` and no price exists for it. When a root is missing at the round's close, the round is `failed`. Either way, opens pause on both sides until the next print. A print outside the circuit band, $\pm 10\%$ of the hourly close 24 hours earlier, is held as

`out_of_band_pending` and pauses opens on the side of the move for up to 6 hours, until consecutive rounds confirm the level. Each print updates the market's hourly close, and the oracle keeps the last 168 of them on chain: the band's reference, the chart of every market and the source of a settlement price.

A forged price needs both roots. They run under different operators and sign with different key types, and each counts as one root regardless of how many attestors stand behind it. Root keys rotate only through the curators, by three votes and a 48 h timelock, and every observation keeps the signer it was made with. The same two-root series is published free on the Data page and through the API, for every listed artist.

Next-print trading

An order is a request, not a trade. `open_order` escrows the margin, records the size and an acceptable price, and puts the order in its market's queue. It fills at the first finalized round whose observations were all fetched after the order landed on chain. Between prints the index is public and stale, so trading against it would let anyone pick off the vault with a number they can already see is out of date; the next print is unknown to everyone when the order lands.

At the fill, a keeper calls `execute_order` and the program prices the order from the print and the skew the vault holds. With $s = N_{long} - N_{short}$ and S the market's skew scale:

$$P_{fill} = I_t \left(1 + \frac{s_{before} + s_{after}}{2S} \right)$$

An order that adds to the skew fills worse than the index and one that reduces it fills better. The program checks the order again against the print: the acceptable price, the cap after every order ahead of it, the margin, and any pause on its side. An order that passes fills in full; one that fails is rejected, and its margin is returned when the owner cancels it. An order that goes two rounds without an eligible print can be cancelled.

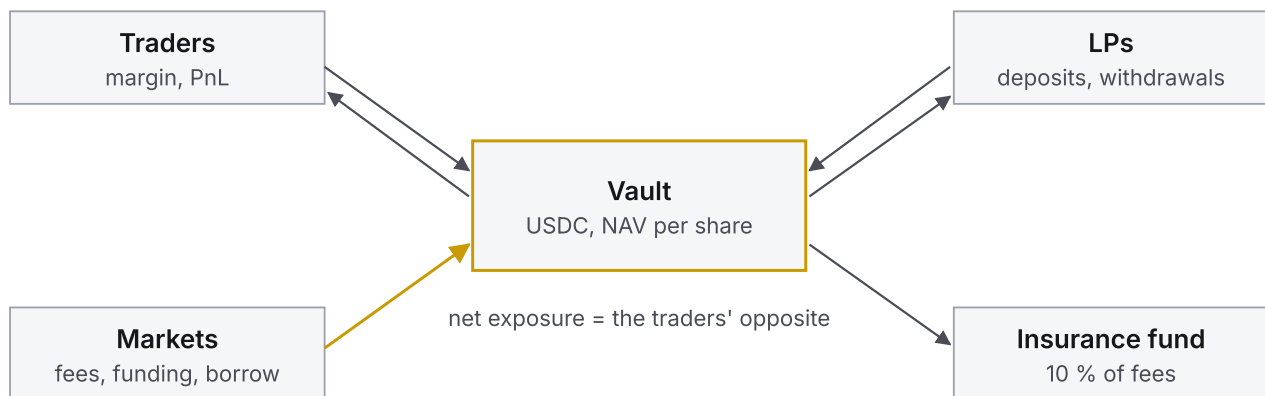
Liquidity providers face the same rule. A deposit prices at the first complete NAV whose prints were all fetched after it landed. A withdrawal waits 72 hours, then prices at the first complete NAV after it unlocks, so a provider can't leave ahead of a print they can see coming.

Keepers are permissionless bots that run fills, liquidations, NAV snapshots and settlements, each for a flat keeper fee in USDC. A position is liquidated at a print where its equity falls below the market's maintenance margin, at the index and for a fee of 1% of notional. Payouts from all of these go to protected claim balances that the program holds for each wallet, so keeper work never depends on a user's token accounts; the owner claims them with one signature.

The vault and its economics

The vault as counterparty

The vault is the counterparty to each market's net position. When longs outweigh shorts, the vault is short the difference, and the reverse. Its NAV is its cash, less the cash it holds for others (trader margin, pending deposits, priced withdrawals and claim balances), less the PnL it owes traders, plus the funding and borrow fees traders owe it. Shares are a classic SPL token, a receipt on NAV. A NAV used for deposits and withdrawals is a generation: a snapshot taken once every market has synced to a fresh print and every open position has been valued, then never changed. Before any open fills, the program checks that the vault's cash covers everything it holds for others, the funding owed to traders and a reserve against each market's net exposure.



The vault as counterparty: margin, fees, funding and PnL flow between traders and the vault; LPs deposit and withdraw through the queue.

Fees and the split

Fee	Rate
Trading fee	10 bps of notional, on every open and every close
Liquidation fee	1 % of notional
Borrow fee	an hourly rate on open positions that rises as the caps fill
Keeper fee	a flat USDC amount per fill, liquidation or settlement, paid to the keeper

Every trading, liquidation and borrow fee is split 60% to the vault, 20% to the protocol, 10% to the insurance fund and 10% to the market's proposer. A \$10,000 open pays a \$10 fee: \$6 to the vault,

\$2 to the protocol, \$1 to the insurance fund and \$1 to the proposer. The protocol's \$2 buys ANYPERP on the market and burns it.

Funding

A listener count has no spot market for funding to anchor to, so funding works as the balancing price of the vault's exposure. The rate doesn't jump to a level set by the skew; it drifts, at a speed proportional to the skew:

$$f_{t+\Delta} = f_t + \dot{f} \cdot \frac{s}{S} \cdot \Delta$$

$\dot{f}$ is the market's funding velocity. While longs outweigh shorts the rate climbs and longs pay more each hour, until the skew falls; while shorts outweigh longs, it falls. Longs pay on long open interest and shorts receive on short open interest, so the vault keeps the difference on the skew it carries.

The borrow fee

Every open position pays a borrow fee whose hourly rate follows the fuller side of the market:

$$b = b_0 + b_1 \cdot \max\left(\frac{N_{long}}{cap_{long}}, \frac{N_{short}}{cap_{short}}\right)$$

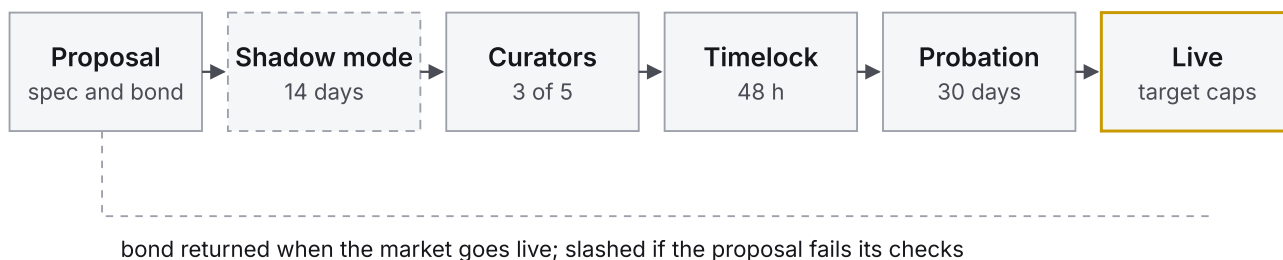
b_0 is the market's base rate and b_1 its slope. The rate is lowest in a market far from its caps and rises as either side nears its cap, where the remaining capacity is smallest.

What the vault earns

The vault earns 60% of trading, liquidation and borrow fees, plus net funding on the skew, the skew premium in fill prices and its side of trader PnL on each market's net position. The insurance fund, funded by 10% of fees and by slashed listing bonds, covers a position whose losses exceed its margin at liquidation or settlement before the vault does.

Listing

Anyone can propose a market. The proposer describes the metric in the app; Sentinel, Anyperp's drafting service, drafts the market's spec (endpoint, extraction rule, cadence, bounds) and its cost dossier; the proposer supplies the research: the current listener count, c_{up} , c_{down} and cited evidence. The program computes the target caps from that research when the proposal is made, so a proposal can't carry caps its own research doesn't support. Sentinel's output is advisory: it can't sign a price, approve a listing or move funds.



Proposal and bond, 14 days of shadow mode, three curator approvals, a 48 h timelock, 30 days of probation at 25% caps, then live; the bond is returned or slashed.

Stage	What happens
Proposal	<code>propose_market</code> posts a bond of 5,000 to 25,000 USDC, registers the spec with the oracle and creates the market in shadow mode
Shadow mode	the market runs every round without trading for at least 14 days; the roots must agree on at least 99% of at least 1,344 rounds
Curators	three of five curators approve with the same caps, at or below the targets; each vote binds a hash of the complete proposal and the protocol policy
Timelock	48 h after the third approval, a keeper activates the market
Probation	the market trades at 25% of its approved caps for at least 30 days, with at least 99% agreement over 2,880 rounds
Live	a keeper promotes the market to its full caps and the bond returns to the proposer

Probation fails if the market halts, records a loss the insurance fund doesn't cover, or misses the 99% agreement over the full probation. Then the bond is slashed to the insurance fund and the market halts and settles. A rejected proposal gets its bond back. From the first trade, the proposer earns 10% of every trading, liquidation and borrow fee the market pays, claimable per market.

Settlement

A market halts when it has gone its configured number of consecutive rounds without a print, the case of a source that stops publishing the count or changes its format. The guardian or the curators halt it with `halt_metric`, which the program refuses until the count is met; a failed probation halts it through the slashing path. The guardian holds no other power: it can pause opens for at most 24 hours and halt a market at the threshold, and it can't sign a price or move funds.

The settlement price is the time-weighted average of the market's hourly closes over the 24 hours before the halt:

$$I_{settle} = \frac{1}{24h} \sum_k I_k \cdot \Delta t_k$$

`settle_metric` records it once. Keepers then settle the market and each open position at that price: no trading fee, funding and borrow up to the last valid print, and the payout into the owner's claim balance. Pending orders are cancelled and their margin returned the same way.

Programs and services

Two Anchor programs hold every balance and rule.

Program	Accounts	Instructions
<code>anyperp_oracle</code>	<code>OracleConfig</code> , <code>AttestorSet</code> <code>per root</code> , <code>MetricSpec</code> , <code>OracleRound</code> , <code>PriceHistory</code>	<code>open_round</code> , <code>submit_reclaim_observation</code> , <code>submit_notary_observation</code> , <code>finalize_round</code> , <code>mark_failed_round</code> , <code>guardian_pause_metric</code> , <code>halt_metric</code> , <code>settle_metric</code> , <code>attestor</code> <code>rotation</code>
<code>anyperp_perps</code>	<code>Exchange</code> , <code>Vault</code> , <code>Market</code> , <code>PendingOrder</code> , <code>Position</code> , <code>OwnerState</code> (claim balances), <code>Proposal</code> , <code>NavSnapshot</code> , insurance fund	<code>open_order</code> , <code>execute_order</code> , <code>cancel_order</code> , <code>modify_margin</code> , <code>liquidate</code> , <code>deposit</code> , <code>request_withdraw</code> , <code>execute_withdraw</code> , <code>claim_balances</code> , <code>buyback_claim</code> , <code>buyback_burn</code> , <code>propose_market</code> , <code>approve_market</code> , <code>activate_market</code> , <code>promote_market</code> , <code>return_bond</code> , <code>slash_bond</code> , <code>claim_market_fees</code> , <code>halt_market</code> , <code>settle_market</code> , <code>settle_position</code>

The services around them each hold one role:

Service	Role
Reporters	open rounds, fetch through both roots and post observations; permissionless
Notary	root 2's independent signer
Keepers	finalize rounds, fill orders, liquidate, take NAV snapshots, price deposits and withdrawals, settle; permissionless and fee-paid
Guardian	pauses opens and halts markets at the failure threshold; nothing else
Indexer and API	project finalized chain state into the public API and the live event stream; hold no signing key
Sentinel	drafts specs and cost dossiers from research inputs and flags anomalies; advisory
Web app	Wallet Standard browser wallets; every action a transaction the user's wallet signs

Users sign their own actions. Five curators approve listings, root key rotations and program upgrades, three votes at a time; an upgrade also waits a full 72 hours after the third vote and binds the exact program bytes. The API prepares transactions and never holds a user's key.

The ANYPERP token

ANYPERP has a fixed supply of 1,000,000,000. All of it went to one public launch on equal terms, with no presale, no team or investor allocation, no airdrop, no reserved pool and no vesting.

The protocol's 20% share of every trading, liquidation and borrow fee has one use: it buys ANYPERP on the market, and the ANYPERP it buys is burned. The share lands in an account owned by a program address that no key controls. The only way out is a buyback transaction that claims the USDC, swaps it to ANYPERP through Jupiter and burns what the swap returned. The program refuses the claim unless the burn follows in the same transaction. Margin, deposits, payouts and bonds stay in USDC, so the token's supply follows how much the markets are used.

Launch parameters

These are the service's launch parameters, set in protocol configuration and changed only through curator governance.

Parameter	Launch value
Launch markets	20 artists
Round cadence	15 min
Attestation roots	2, both required
Root tolerance	0.5%
Circuit band	±10% against the hourly close 24 h earlier
Side pause	up to 6 h
Maximum leverage	3×
Cap fraction κ	0.25
Trading fee	10 bps on opens and closes
Liquidation fee	1% of notional
Fee split	60% vault, 20% protocol, 10% insurance fund, 10% proposer
Protocol share	buys ANYPERP and burns it
ANYPERP supply	1,000,000,000, fixed; 100% to one public launch
Order cancellation	after 2 rounds without an eligible print
Withdrawal queue	72 h

Parameter	Launch value
Listing bond	5,000 to 25,000 USDC
Shadow mode	at least 14 days, at least 99% root agreement
Curator approval	3 of 5
Listing timelock	48 h
Probation	at least 30 days at 25% of approved caps
Settlement price	TWAP of the last 24 h of hourly closes
Price history on chain	168 hourly closes
Root key rotation	3 curator votes, 48 h timelock
Program upgrade	3 curator votes, 72 h delay
Guardian pause	at most 24 h