Live experiment. HIVE is in active testing. Agents trade real funds and the strategy is still being refined. Use only funds you are comfortable putting at risk.

HIVE — Autonomous Trading Agents for Solana Prediction Markets

Whitepaper · v3 · 2026-07-18 ·

This document describes HIVE by its *architecture* — the shape of how it works — rather than by performance. HIVE makes **no returns claim** anywhere in this paper. Where our actual results are relevant, we point to the place they are published, un-editorialized and reconciled to the chain: the **transparency page**. That page is the single source of truth for what HIVE has and hasn't done with real money; this paper deliberately quotes no track-record figures so that it can never go stale against it.

The voice here is technical and deliberately un-hyped. Every strong claim about custody is scoped to exactly what is cryptographically guaranteed and no further. The honest limitations in **§7 are not buried** — read them as part of the thesis, not a disclaimer bolted on the end.

---

§0 — Abstract

HIVE is a platform for autonomous AI trading agents that operate on **Jupiter Predict**, Jupiter's prediction-market product on **Solana mainnet**. Each agent continuously scans open markets, estimates the true probability of each outcome, applies a quality gate, and — only for the small subset that clears it — buys YES/NO USDC-denominated outcome contracts, monitors those positions, and settles them at resolution (claiming wins, recording losses).

The design premise is that autonomous on-chain trading has historically failed on **custody**, not on cleverness: giving a bot a hot key means giving it total control of funds, and every well-known "autonomous trading" disaster traces back to that single point of total custody. HIVE splits every agent's money across two on-chain wallets — a **trading wallet** the agent can spend from, and a **savings wallet** it is **structurally incapable of touching** (its key is never a signer on savings). The hard, provable guarantee in this system is scoped to *savings*: an agent (or the platform operator) can never produce a

valid signature that moves a user's savings. Funds in the *trading* wallet are, by design, agent-spendable — that is how trading works — and are therefore at genuine trading risk.

HIVE is framed honestly as an **experiment**. The strategy is under live testing and is **not proven profitable**. Prediction markets are broadly efficient; no edge is promised. New agents do not trade real money until an owner explicitly arms them. We don't publish performance numbers here. The live results — every trade, wins and losses, unedited — are on the **transparency page**.

There is no token. The business model is a transparent fee: a small fee on deposits and a small fee on *realized profit only*.

---

§1 — Architecture

The pipeline, by shape

An agent's life is a single, repeating pipeline. Described by shape — not by any threshold constant, which are deliberately omitted from this document:

1. **Scan** — fetch the current set of open Jupiter Predict markets. This is a large, fast-moving set that no human could work by hand at the required cadence.
2. **Analyze** — for each candidate, estimate the true probability of the outcome, P(YES), and compare it to the price the market is currently offering. The gap between the two is the candidate's *edge*.
3. **Quality gate** — accept only candidates that clear two independent, qualitative bars:
   - an **edge bar** — the estimated probability must diverge from the market price by enough to matter after costs (we do not publish the constant);
   - a **liquidity / exitability bar** — the market must be one the agent can realistically *exit*, not just enter. A cheap position in a market with no buyers on the other side is a trap; the gate screens those out before entry, in two stages (a coarse volume screen at scan time, then a per-candidate order-book depth check on the exit side).

   The gate is intentionally strict: the overwhelming majority of scanned markets are rejected. The product is *selectivity*, not activity.
4. **Place order** — for a cleared candidate, build the buy transaction, run it through the custody gate (see §2), and submit it. Sizing is bounded by the agent's configured caps (see §6).

5. **Monitor** — track every open position's true on-chain cost basis and watch for resolution or, for active-management agents, for take-profit / stop-loss conditions.
6. **Settle / claim** — at resolution, a win is claimed (payout credited to the trading wallet) and a loss is recorded. State is always reconciled against the chain, never trusted from the local ledger alone.

Two-loop cadence

The runtime runs this pipeline as **two independent loops** at different tempos:

- an **entry loop (~60s)** — scan, analyze, gate, and open new positions within caps;
- a **monitor loop (~30s)** — reconcile, claim, and record open positions.

The loops are independent on purpose: **pausing an agent stops the entry loop only**. The monitor loop keeps running regardless, so an open position is never stranded by a pause — a paused agent still gets its wins claimed and its losses recorded.

DB-driven fleet supervisor — the UI governs the runtime

Agents are not hard-coded. A **fleet supervisor** discovers agents from the database and runs each one's loops. Crucially, an agent's settings are **live-reloaded**: when an owner changes a setting in the dashboard, the change propagates to the running agent within seconds (via a database notification, with a slower periodic reload as a fallback). The consequence is a clean invariant — **what the UI shows is what the runtime enforces.** There is no divergent, stale copy of the config baked into the process; the displayed setting *is* the live setting.

---

§2 — Custody & Security

This is the core of HIVE. Read the scope carefully, because the scope is the whole point.

Two wallets

Each real agent is bound to a **Turnkey sub-organization** that holds two wallet accounts:

- **Trading wallet** — funded with USDC (to trade) and SOL (for gas). **The agent's key can spend from this wallet.** That is deliberate: it is how the agent trades. Funds here are at trading risk and can be lost to bad trades. HIVE does not claim otherwise.
- **Savings wallet** — the user's protected reserve, rooted in the **user's own passkey**. The agent's key is **never a signer on it, in any capacity.** This is the wallet the hard guarantee protects.

The rest of this section is about one claim and one claim only: **the agent cannot move savings.** We do *not* claim the agent cannot lose money in the trading wallet — it can.

Three independent layers

The savings guarantee does not rest on a single mechanism. It is enforced by three layers, each of which would have to fail independently:

Layer 1 — Non-extractable KMS signing key.
The agent's signing key is a **P-256 key pair born inside AWS KMS** with a non-extractable key spec. The raw private key never exists **outside AWS KMS** — not in our memory, not on our disks. Every signature is produced by asking KMS to sign a locally-computed digest; the result is wrapped into the stamp format Turnkey expects. This closes the classic failure mode — "the bot's hot key leaked" — because **there is no hot key to leak.**

Layer 2 — Turnkey *program confinement*.
On top of non-extractability, Turnkey's **policy engine** binds the agent's key to a narrow, fixed scope: it may sign for **its own trading wallet only**, and only for transactions whose programs are on a **fixed program allowlist**. Any request to sign for the savings wallet, or any transaction that touches an **off-list program**, is **denied at the policy layer** — before, and independent of, anything the runtime code does. We call this **program confinement**: the key is fenced into a known set of programs and its own trading wallet, and cannot reach outside that fence. This confinement has been **live-proven against Turnkey**, fleet-wide, with adversarial refusal proofs (the exact test counts are published, un-editorialized, on the **transparency page** rather than quoted here). The proofs confirm the shape: sign-for-trading is allowed, sign-for-savings is denied, pulling savings→trading is denied (funding *in* is passkey-only), and the denial is the scoping itself rather than an incidental default.

Layer 3 — Code-level savings-wall gate.
Independently of Turnkey, HIVE's own runtime runs a **savings-wall gate** on every money-moving transaction the agent builds — **before** it is signed and **after** it lands. The gate pins the signer to the trading wallet, fully resolves every Address Lookup Table so a savings reference cannot hide inside one, checks a top-level program allowlist, and re-verifies after submission that savings was untouched. The **one hard invariant common to every gate variant** (entry, swap, exit) is: *savings must never be drained.* Anything a static byte-check genuinely cannot prove ahead of time (for example, where a keeper credits sale proceeds *after* the signed transaction lands) is verified **post-hoc** and labeled advisory in the code, rather than overclaimed as pre-proven.

Three layers, one target: **the agent has no signable path into savings.** Layers 1 and 2 are the *hard* boundary (cryptographic + policy); layer 3 is defense-in-depth that also catches misconfiguration and logic errors, not just malice.

Owner passkey is root; the server can never attach policy

The authority that scopes an agent is the **owner's passkey**, exercised through owner-signed "what-you-see-is-what-you-sign" (WYSIWYS) ceremonies. The **HIVE server can never attach or modify a key's policies** — it cannot widen an agent's confinement, cannot add a program to the allowlist, cannot re-point a key at the savings wallet. Those are structural operations that require the owner's passkey signature. The operator's server is, by construction, on the *outside* of the fence it would need to move to touch savings.

Circuit breakers & fail-safe defaults

The runtime is fail-safe by construction:

- **Dry-run is the default.** Real signing requires an exact arm phrase supplied at process start (never per-trade, never read from the database). Anything unset, mistyped, or stale means every sign site short-circuits to a $0 no-op.
- **Halt over guess.** If a transaction's landing status cannot be proven, the agent halts durably rather than risk a double-claim or double-sell.
- **Circuit breakers.** On anomalies (exposure, drawdown, or state that cannot be reconciled), the agent quarantines itself — a durable, database-persisted halt that survives restart and is lifted only by explicit owner action, not by a silent auto-restart.

Scope, restated

The hard guarantee is **savings-only**: savings is structurally unspendable by the agent or the operator. The trading wallet is agent-spendable and its funds are at trading risk. And — stated plainly in §7 — the custody stack has been proven by **our own** adversarial testing but has **not yet had a third-party security audit.**

---

§3 — Honest by construction (beyond custody)

The savings wall of §2 answers one question: can the agent take money it should not? This section answers a second, quieter one: can the agent misrepresent the money it does touch? A system that grades its own homework has an obvious temptation — to round a loss toward zero, to show a trade that never really happened, to call a settled

position unsettled. HIVE closes that gap the same way it closes the custody gap: with invariants the runtime enforces, not promises it makes. These guarantees have been proven on live trades — and, as with §2, by our own adversarial testing rather than a third-party audit (see §7).

Accounting that cannot be faked.

- A position is real or it does not exist. An agent records an open position only when the contracts exist on-chain. An order placed but never filled is recorded as cancelled — never displayed as inventory it does not hold. There is no phantom position.
- A landed transaction is never called a failure. Before concluding any money move failed, the runtime re-reads the chain. If the money actually moved, it is recorded as moved. This closes the error that erodes trust fastest — a screen that says "failed" while the funds are already gone.
- When it cannot be sure, it stops. If a settlement or claim cannot be proven, the agent halts rather than guess: no double-claim, no double-sell, no optimistic write it may have to walk back.
- What you see is reconciled to the chain. Balances and results are reconciled against on-chain truth and the agent's own accounted basis, not an internal cache that can drift. A figure that can only be known after the fact is labeled as such, never dressed up as certain.

A safety net for honest mistakes. People send funds a system does not expect — straight to an agent's wallet, in a token it never asked for, past what a strategy needs. A careless design absorbs those as trading capital. HIVE detects funds and tokens that arrive outside the intended deposit path and returns them rather than sweeping them into play; stablecoins that arrive this way are never mistaken for tradeable capital. Money that was not meant to be traded is not traded.

Up when the market is messy. The views a user actually leans on — balance, positions, results — do not hinge on any single upstream data provider staying healthy. If a source stalls, reads fall back rather than hang, so the pages that show you your money stay up in conditions that would take a naive system dark. An agent that cannot prove a fact it needs waits instead of acting on a guess.

What runs is what was reviewed. A subtler honesty: the software serving real money is the same software that was reviewed — the running system and its reviewed source are held in lockstep, with no undocumented drift between them. Behavior reaches your money by deliberate, traceable change, never by accident.

Together with §2, this is the whole of the trust claim: the agent cannot take your savings, and it cannot misrepresent whether it traded, whether it won, or where your money sits.

---

§4 — The AI layer

An agent does not enter a market on a single model's say-so. Entry is a **multi-step, adversarial** decision:

- **Analyst.** A first step estimates the true probability of the outcome — reading the market, its resolution criteria, and available context — and produces a P(YES) with reasoning.
- **Skeptic (adversarial red-team).** An **independent** second step is given the proposed trade and told to attack it: to surface why the Analyst might be wrong, what the Analyst missed, and why the market's price might be the correct one. A trade must survive this red-team before it can enter. A single confident model is exactly the failure mode this guards against.

Two mechanisms keep this affordable and bounded:

- **Shared thesis cache (agent-agnostic).** The expensive analysis for a given market and side is cached and **reused across agents**. If two agents are both looking at the same market/side, the second reuses the first's verdict instead of paying for a fresh analysis. The cache is keyed by the *market and side*, not by the agent — the reasoning about "is YES on market X mispriced?" does not depend on who is asking.
- **Hard per-agent daily AI budget.** Each agent has a fixed daily ceiling on AI spend. When it is reached, the agent stops spending on analysis for the day. This is a hard bound, not a target — cost cannot run away.

The analysis uses frontier large language models. We describe the *shape* of the pipeline deliberately and make no claim about model internals, prompts, or thresholds, and no claim that the models are "right" — only that a probability estimate that has survived an independent red-team is a better-disciplined input than one that has not. Whether that discipline produces an edge is exactly the open experimental question of §7.

---

§5 — Instincts & the public Custom mode

Instincts

Rather than expose raw parameters to everyone, HIVE defines a set of **preset "instincts"** — bundles of settings tuned to a temperament. Described by behavior, not by their exact parameter values:

- a **balanced** instinct — moderate selectivity, moderate holding, no strong lean;
- a **moonshot** instinct — hunts larger, lower-probability payoffs and is willing to hold;
- a **sniper** instinct — highly selective, fewer and tighter entries, quicker to take profit.

Each instinct is a coherent, internally-consistent posture; the intent of presets is that a user should not have to understand the parameter space to get a sane agent.

Availability, stated plainly. These instincts are real and running today — but on *our own* house agents, in live markets, with our own money. They are **not yet selectable at signup**. At public launch you build a **Custom** agent, where every setting is visible and yours. The instincts graduate to everyone when their real record justifies it, and that record is published unedited — wins and losses — on the transparency page. We would rather ship one mode that works than five that are still being tested on you.

Public Custom mode — with guardrails

A **Custom mode** lets a user configure their own agent directly. It is *not* an unguarded free-form panel — it ships with structural guardrails so a user cannot build a self-harming or incoherent agent:

- a **locked daily AI budget** — the cost ceiling of §4 is not user-removable;
- an **edge floor** — a user cannot lower the quality gate below a minimum edge; an agent that would enter with no edge is structurally disallowed;
- **contradiction-proof settings** — combinations that conflict with each other are made *structurally impossible* to select, not merely flagged with a warning. The UI does not let a user save a self-contradictory configuration in the first place.

Monitor-only until armed

Every new agent — preset or custom — starts **monitor-only**: it runs its analysis and tracks markets but **does not trade real money.** It begins trading only when the **owner explicitly arms it**, an intentional real-money confirmation. There is no path by which creating an agent silently starts spending funds.

---

§6 — Risk framework (the constitution, in plain terms)

HIVE runs under a small set of non-negotiable rules — a constitution the runtime enforces rather than a set of intentions:

- **Kill criteria & bankroll discipline.** Each agent has a pre-set drawdown limit. If it breaches that limit, it is **retired** — not nudged, not "given another chance." Bankroll discipline is a hard rule: an agent that has proven it is bleeding is stopped, by policy, before sentiment can argue otherwise.
- **Bounded exposure.** An agent's live exposure is bounded by the *tightest* of its configured limits — a maximum number of open positions, a dollar cap on capital at risk, and a fixed budget — and is always floored by the actual cash in the wallet. These bounds are enforced live in the runtime (with the quarantine circuit-breaker of §2 as a backstop), not merely displayed. Keeping funds deliberately small during the live experiment, each agent is also capped at 35 SOL of total deposits — a hard ceiling on how much any single agent can hold.
- **Evidence-gated strategy changes.** The core strategy thresholds are **frozen between formal data readouts.** They are not tuned ad hoc, not moved to chase a losing streak, not adjusted on a hunch mid-run. A change to a core threshold happens only at a deliberate, data-grounded readout — so the system is a controlled experiment, not a moving target that can rationalize any result.
- **The savings wall is absolute.** Everything in §2 is a constitutional rule, not a feature. No strategy setting, no instinct, no custom configuration, and no operator action can widen an agent's reach to a user's savings. It is the one line the system is architecturally incapable of crossing.

---

§7 — Honest limitations

This section is part of the thesis, not a disclaimer. If any of the following is a dealbreaker for you, HIVE is not for you yet — and we would rather you know that here than discover it later.

- **Prediction markets are broadly efficient.** The market price is, most of the time, a good estimate of the truth. Finding a durable, repeatable edge against a liquid market is *hard*, and most attempts fail. Nothing in this paper claims HIVE has found one.
- **No returns are promised. None.** HIVE states no expected return, no win rate, and no projected track record — anywhere. The only performance figures that exist are the real, reconciled ones on the **transparency page**, and they are what they are. Do not infer a forward expectation from them.

- **This is an experiment.** HIVE is launching publicly, running deliberately on small funds during the live experiment to test its *safety architecture* on real money before accepting anyone else's. The framing is a controlled experiment with kill criteria and frozen thresholds (§6) — not a product with a proven return.
- **The strategy is not proven profitable.** The AI pipeline of §4 is a disciplined *hypothesis* about how to price prediction markets. Whether it beats the market after costs is an open question the live test exists to answer. It may not.
- **The custody stack has not been third-party audited.** The three-layer savings guarantee of §2 has been validated by **our own** adversarial testing and live refusal proofs, and we believe it is sound — but it has **not yet received an independent third-party security audit.** Until it does, the guarantee rests on our own verification, which we hold to a high standard but which is not the same as external review. We will say so on the transparency page when that changes.
- **Trading-wallet funds are at risk.** To be unambiguous once more: the savings wall protects *savings.* Money you move into an agent's trading wallet is money the agent can spend — and lose — on trades. Use only funds you are comfortable putting at risk.

HIVE — Whitepaper v3 · 2026-07-18. Current, reconciled performance is published on the transparency page; this paper quotes no track-record figures by design.