AlphaEngine Overview

1) Hook with Private Swaps

Role: privacy and last-mile execution.

- Users deposit to the HookVault. Balances are encrypted (eUSDC, eUSDT).
- Users submit encrypted swap intents.
- AVS batches, nets, and posts bounds.
- Hook executes only the net AMM trade, then applies encrypted settles.

  Outcome: no per-user linkage, MEV minimized, only aggregate size hits the pool.

2) Pool Liquidity to BoringVault with Veda-style constraints

Role: capital firewall and policy.

- Liquidity from the pool side routes into a BoringVault that enforces a Merkle-proof policy tree (Veda Vault pattern).
- Tree encodes: allowed assets, DEXes, venues, position caps, slippage, price oracles, TTL, exposure limits, kill-switches, operator lists.
- Every execution path must verify a proof against this tree.

  Outcome: even if a trader or executor is compromised, funds cannot exit approved constraints.

3) Arena: ladder to curate top traders

Role: talent discovery and curation, not forced auto-execution.

- Bottom tier: learn and simulate.
- Middle tier: trade battles to level up.

- Top tier: a curated set of traders get “submission rights” to propose live deployments for the vault.

  Important: CiFHErTradeArena was a proving ground. It showed auto-execution per epoch is possible but not optimal. In production the Arena selects traders who earn the right to submit, not the right to auto-execute.

4) AVS has two distinct duties

a) **Batcher for the Hook swaps**

- Decrypt with permits, simulate, net, compute bounds, aggregate signatures, and gate the Hook.
- Goal: break user linkage and constrain execution to safe bounds.

b) **Guardian for strategy submissions**

- For Arena simulations: run sims and pick winners. No capital moves.
- For top-trader submissions: simulate proposed trades against the Veda policy and risk limits. Accept or reject.
- Enforce per-trade and per-epoch risk envelopes. Slashing on equivocation, stale snapshots, or bounds violations.

How it fits together: lifecycle

```
flowchart LR
  U[Users] -->|deposit, encrypted intents| H(HookVault)
  H -->|TaskCreated| AVS1[AVS: Batcher]
  AVS1 -->|bounds + quorum| H
  H -->|net swap only| AMM[Uniswap v4]
  AMM --> H -->|encrypted settles| U

  subgraph "Arena Ladder"
    L1[Learn tier] --> L2[Battle tier] --> L3[Top traders]
  end
  L3 -->|propose trades| GOV[Submission Router]
  GOV -->|Merkle proof checks| VEDA[Veda Policy Tree]
  VEDA --> AVS2[AVS: Guardian]
  AVS2 -->|simulate + validate| VEDA
  AVS2 -->|accept/reject| BV[BoringVault]
  BV -->|approved calls only| DeFi[Venues: AMMs, Aave, etc.]
```

Control model and guardrails

- **Separation of powers**:

  Hook privacy and batching is orthogonal to strategy selection.

- **Veda constraints**:

  Every live trade must present a Merkle proof. No proof, no spend.

- **AVS consensus**:

  Batcher signs bounds for swaps. Guardian signs accept/reject for strategy calls. Both are slashable.

- **No forced execution from Arena**:

  Arena ranks skill. Only top traders gain the right to submit. AVS still blocks anything outside policy or risk.

Key advantages

- Privacy by default for swappers.
- Market-driven alpha without leaking alpha.
- Defense in depth: Veda policy tree + AVS Guardian + slashing.
- Clear talent pipeline: learn → battle → curated submitters.

If you want, I can turn this into a one-page architecture sheet or a slide with a clean diagram and the exact interfaces to drop into your repo.