Hyperliquid 技术深度解析：从链上订单簿到机构级风险引擎的全景技术图谱

背景介绍

过去五年，去中心化衍生品市场经历了从试验、爆发到再平衡的周期。从 2020 年到 2022 年，基于 **AMM（Automated Market Maker）** 的永续合约协议（如 **Perpetual Protocol**、**GMX**）占据了市场的主导地位，凭借低门槛的流动性模型和快速上线能力，成为 DeFi 用户的第一选择。然而，随着市场的成熟和专业交易需求的快速增长，**CLOB（Central Limit Order Book，中心化限价订单簿）** 再次成为链上交易架构的焦点——尤其是在 **应用链（Appchain）化** 的趋势推动下，链上 CLOB 永续协议正快速崛起。

Hyperliquid（以下简称 **HL**）便是在这一浪潮下的产物，它通过一套高性能、低延迟、模块化的链上撮合和清算体系，试图将 **中心化交易所（CEX）的深度和效率** 带入去中心化世界。

研究动机：CLOB 永续的回归与 Appchain 化趋势

AMM 永续协议的成功证明了链上衍生品的巨大市场需求，但也暴露出多个 **架构性的局限**：

- **价格发现滞后**：AMM 的定价基于资金池曲线（vAMM 或 GLP），无法与高频波动市场同步，导致滑点和资金费率失真。
- **深度受限**：池子深度取决于 LP 的被动资金，无法满足机构交易或大额策略订单的冲击需求。
- **风险管理不足**：AMM 模式难以**实现高精度的逐仓风控**、组合**保证金或专业的清算体系**。

与此同时，链上基础设施的演进（如 **高性能 L1/L2**、**模块化 Rollup**、**专用应用链框架**）使得 **低延迟、高并发的链上撮合引擎** 成为可能。链上 **CLOB 模式** 再度进入行业视野，并被视为衍生品赛道下一阶段的核心基础设施。

Hyperliquid 选择在这一赛道中深耕，结合专用链架构和高性能撮合引擎，构建出能够承载机构量化和高频策略的链上环境。这种设计不仅满足了专业交易者的需求，也为链上生态引入了全新的技术范式。

HL 的定位：高性能链上 CLOB 永续 + 现货一体化

Hyperliquid 并非简单的永续协议，而是一套集 **现货市场** 和 **永续合约市场** 于一体的完整交易基础设施。

- **专用链架构（Appchain）**
- HL 运行在专用链上，避免了以太坊主网或通用 L2 的拥堵和性能瓶颈，能够做到亚秒级的撮合延迟。
- **原生链上订单簿（On-chain CLOB）**
- 所有订单、撮合和状态变更完全链上化，保证交易透明性与可验证性。
- **统一保证金账户**
- 跨现货和永续产品的组合保证金设计，提高资金使用效率。
- **低延迟、高并发**
- 高性能引擎支持高频交易策略，与传统 CEX 体验接近。
- **机构级风控体系**
- 支持逐仓/全仓模式、实时风险计算、自动清算和保险基金管理。

凭借这一架构，Hyperliquid 被视为下一代链上衍生品基础设施，将 CEX 的体验与 DEX 的透明性和可组合性结合起来。

与 AMM 型永续的根本差异

Hyperliquid 与 AMM 型永续协议的差异，可以从三个技术维度拆解：

1. 价格发现机制

- **AMM**：基于池子的资金曲线进行被动报价，价格调整依赖套利者与市场交易，存在 **时滞** 和 **偏差**。
- **CLOB**：由订单簿驱动的实时价格形成机制，撮合与深度直接反映市场供需，实现与中心化交易所同频的价格发现。

---

2. 流动性深度

- **AMM**：深度依赖 LP 的静态资金，无法满足机构或大资金的需求。
- **CLOB**：通过主动挂单和撮合机制，聚合自然深度，支持更复杂的策略和更紧密的价差。

---

3. 风险管理

- **AMM**：风险模型简单，以池子均摊损益为主，缺乏逐仓和组合保证金管理。
- **CLOB**：支持逐仓、全仓和组合保证金架构，清算逻辑细致，能够承载机构级的风控需求。

这种从 **被动曲线撮合** 到 **主动订单簿匹配** 的架构跃迁，使得 Hyperliquid 不仅具备专业交易的性能优势，也拥有 AMM 难以匹敌的可扩展性和安全性。

为了系统解析 Hyperliquid 的技术体系，本文将按照 **从宏观到微观、从理论到实践** 的逻辑展开：

1. **系统总览**：解读 HL 的架构蓝图与核心组件，展示从用户下单到链上撮合的完整链路。
2. **共识与执行**：深入解析高吞吐共识机制与链上状态机设计。
3. **订单簿与撮合引擎**：剖析订单数据结构、撮合逻辑、延迟优化与公平性机制。
4. **永续合约机制**：从资金费率计算到 PnL 记账的数学与实现细节。
5. **风险引擎与清算体系**：详解逐仓、全仓、ADL 与保险基金的风险防控体系。
6. **预言机与价格基础设施**：多源聚合与异常保护机制。
7. **账户模型与结算记账**：统一保证金架构及资金流转逻辑。
8. **MEV 与排序公平性**：链上交易安全与反抢跑设计。
9. **安全工程与合规**：合约安全、参数治理与合规接口。
10. **可观测性与运维**：指标、日志、监控与应急机制。
11. **扩展方向**：产品进化与新市场（如 RWA、期权）的探索。
12. **实证对比与实验**：性能测试、极端场景压力测试与竞品对比。
13. **开发者指南**：API 接入、策略开发与仿真工具。
14. **风险揭示与治理**：从工程视角看治理、透明度与行业挑战。
15. **总结**：回顾 HL 的技术演化与未来路径。

---

1. 系统总览：控制面与数据面

Hyperliquid（HL）的架构充分体现了 **专用链（Appchain）+ 高性能 CLOB 引擎** 的融合理念。它既保留了链上执行的透明性和安全性，又通过定制化链下优化与模块化链上状态管理，实现了低延迟、高吞吐的撮合体验。本节将围绕 **架构组件、关键交易路径、性能指标** 三个维度，拆解 Hyperliquid 的技术蓝图。

1.1 组件总览

Hyperliquid 的架构可以分为 **控制面（Control Plane）** 和 **数据面（Data Plane）** 两层。

（1）客户端 / SDK 与 API 网关

- **SDK 支持**
- 提供 TypeScript、Python、Rust 等多语言 SDK，便于交易所、策略团队或量化机器人接入。
- **API 网关**
    - **REST API**：用于下单、撤单、资产查询等低频指令。
    - **WebSocket API**：实时订阅订单簿（Level 2）、逐笔成交、资金费率和风险事件。
- **签名与鉴权**
    - 所有请求均需私钥签名，保证消息不可篡改。
    - 支持 HMAC 和链上地址签名两套方案。

（2）Sequencer / Validator（排序与出块）

- **Sequencer**
    - 负责接收客户端订单请求并进入 **Mempool** 队列。
    - 按时间优先或批次优先规则进行排序。
    - 确保撮合的公平性，防止抢跑（Front-running）。
- **Validator**

- 将 Sequencer 打包的交易区块进行验证并广播到全网节点。
- 提供快速确认（Soft Finality）和最终确认（Hard Finality）机制。

（3）Matching & Risk Engine（撮合与风险引擎）

- **撮合引擎**
  - 完全链上记录订单簿状态，但执行路径做极致优化。
  - 采用 **价格优先、时间优先（Price-Time Priority）** 的排序逻辑。
  - 支持多种订单类型：限价单、市价单、止盈止损单、**IOC/FOK、Post-Only**。
- **风险引擎**
  - 实时计算账户净值、保证金比例、资金使用率。
  - 在每次撮合前进行资金与杠杆校验，确保订单有效。

（4）Settlement / State（结算与状态存储）

- **统一保证金账户**
- 跨现货与永续共享账户权益，支持多资产作为抵押品。
- **状态存储**
  - 采用高性能 Key-Value 存储结构（类似 RocksDB/LevelDB）进行链上同步。
  - 状态哈希写入区块链，保证可验证性与审计性。

（5）Oracle Aggregation（多源预言机汇聚）

- 汇聚 **Pyth、Chainlink、Kaiko** 等多源报价。
- 使用 **中位数 + TWAP + 异常剔除** 算法生成指数价（Index Price）和标记价（Mark Price）。
- 定期心跳（Heartbeat）与延迟监控，确保报价稳定可靠。

（6）Liquidation & Insurance（清算与保险基金）

- **清算引擎**
  - 实时监控账户健康度（Margin Ratio）。
  - 触发逐仓或全仓清算，将仓位以最优价格快速平仓。
- **保险基金**
  - 用于兜底极端行情下的穿仓损失。
  - 资金规模和使用策略链上透明公开。

（7）Indexer / Analytics（可观测性与分析）

- **Indexer**

- 将链上状态与撮合事件索引化，供前端和分析工具使用。
- 支持按时间戳或交易 ID 快速查询。

- **Analytics**

  - 提供实时监控指标：交易量、深度、资金费率、清算记录。
  - 通过 Grafana / Prometheus 等工具可视化。

1.2 关键交易路径

Hyperliquid 的完整交易链路可以用下列步骤概括：

1. **下单（Client → API）**

   用户或策略机器人通过 SDK/REST/WebSocket 提交订单请求，并进行私钥签名。

2. **排队（Sequencer → Mempool）**

   Sequencer 接收订单，进入队列并按时间/批次排序。

3. **撮合（Matching Engine）**

   撮合引擎根据订单簿深度撮合成交，更新账户余额和仓位信息。

4. **记账（State Update）**

   撮合结果同步到链上状态存储，生成状态哈希写入区块。

5. **资金费率计算（Funding Rate）**

   风险引擎根据指数价与市场价差计算资金费率，按时计提。

6. **清算/回收（Liquidation）**

   当账户保证金不足时，触发清算，将仓位平仓并更新账户权益。

7. **回执返回（Client）**

   用户通过 WebSocket 或 REST 接口接收撮合结果和交易回执。

1.3 性能目标与关键 SLO

作为高频衍生品交易平台，Hyperliquid 的性能和稳定性是核心竞争力。其 **服务水平目标（SLO）** 主要包括以下指标：

指标	目标值	说明
撮合延迟	< 500ms	单笔订单从下单到成交确认的时间
区块出块时间	~1 秒	Sequencer 快速排序并提交
状态同步延迟	< 2 秒	区块状态写入链上账本的时间
吞吐量（TPS）	5,000+ TPS	高峰期撮合支持
可用性	99.95%+	高可靠性架构，避免交易中断
恢复时间（RTO）	⩽ 5 分钟	极端故障下快速恢复服务

通过这些指标，Hyperliquid 能够在 **低延迟、高稳定性** 与 **链上可验证性** 之间找到平衡。

1.4 架构特点总结

- **高性能链上撮合**：定制化链层优化，保证高频策略可行。
- **模块化设计**：撮合、风险、清算、预言机、存储完全解耦，便于扩展与升级。
- **透明性与可验证性**：订单、成交、资金流动均链上可查，支持审计与监管。
- **弹性与冗余**：多节点容灾和自动热切换机制，保障极端行情下的连续可用。

---

2. 共识与执行：高吞吐低延迟的链上撮合底座

Hyperliquid（HL）作为一条专用的链上交易基础设施，其性能核心来源于 **高性能的共识机制** 和 **优化过的状态执行引擎**。通过在架构层面专门为 **订单簿撮合（CLOB）** 场景定制底层逻辑，HL 在保证链上透明性和安全性的同时，将撮合延迟降到接近中心化交易所（CEX）的水准。本节从 **共识范式、交易生命周期、确认模型、状态机设计以及异常处理机制** 五个维度，深入解析 HL 如何实现高吞吐低延迟的链上撮合。

2.1 共识范式假设：BFT 与自研快速出块

Hyperliquid 的共识设计是性能优化的基础。它采用 **BFT 类协议（Byzantine Fault Tolerant）** 的改进版本，结合 **快速出块机制**，在吞吐量、延迟与安全性之间取得平衡。

架构特点

- **单主 Leader 模式**：单个 Sequencer 作为 Leader 进行交易排序，其他 Validator 验证并签名确认。
- **低出块延迟**：出块时间约 1 秒，支持亚秒级确认。
- **批处理（Batching）**：同一批交易打包进入区块，提升吞吐效率。
- **可配置共识深度**：在安全性与性能之间动态平衡，如极端行情下可临时增加确认阈值。

这种架构参考了 Tendermint 的实现，但针对高频交易进行了深度定制，以确保 **TPS（交易处理能力）** 和 **链上状态一致性**。

2.2 交易生命周期：从提交到写入链上

交易在进入链上前，需要经历严格的验证与排序过程，确保公平性和确定性。整个生命周期包括以下几个阶段：

Mempool 接收与去重

- 所有来自 REST 或 WebSocket 的订单先进入 **Mempool**。
- 去重策略：
  - 按交易哈希去重
  - 同账户的重复提交检测
- 合法性校验：
  - 签名验证
  - 账户余额/保证金预检查

排序策略

HL 的排序机制结合了 **时间优先** 和 **批次公平**：

- **时间优先（FIFO）**：早提交的订单优先进入排序队列。
- **批次公平（Fair Batch Auction）**：在微批（如 100ms 时间窗）内将订单批量排序，防止高频抢跑。

伪码如下：

```
Code block
1   for each block_window:
```

```
    batch = collect_orders(time_window=100ms)
    sorted_batch = sort(batch, key=(price, time))
    commit(sorted_batch)
```

撮合与状态更新

- 由 Matching Engine 进行价格优先/时间优先撮合
- 撮合结果写入链上状态树（State Tree）
- 返回执行回执给客户端

2.3 确认模型：乐观确认 vs 最终性

高性能 CLOB 链上撮合需要平衡 **速度** 与 **安全**，HL 引入了 **双层确认模型**：

乐观确认（Optimistic Confirmation）

- 撮合完成后立即返回确认
- 平均延迟 < 500ms
- 适用于高频交易和低风险场景

---

8. **最终确认（Finality）**

- 区块在全网 Validator 达成共识后，状态不可逆转
- 通常 2–3 秒内完成最终确认
- 适用于资金转移、清算、风控事件等高价值操作

对订单簿连续竞价的影响

- **高频策略**：乐观确认保证了流畅体验
- **风险敏感策略**：最终确认提供强一致性保障
- **混合撮合**：满足专业做市商对延迟与确定性的双重需求

2.4 状态机设计：幂等与原子性

状态机的设计是撮合引擎稳定运行的保障。

幂等性（Idempotency）

- 每个订单在执行引擎中有唯一 ID
- 重放或重复提交不会导致状态不一致

事务原子性（Atomicity）

- 撮合和账户余额更新作为一个原子事务提交
- 要么全部成功，要么全部回滚，避免状态不一致

伪码示例：

```
begin transaction
  update_orderbook()
  update_account_balance()
commit transaction
```

这种设计确保了极端行情下状态一致性，即使高频下单/撤单也不会引发脏状态。

2.5 异常与回滚机制

在去中心化环境中，链的 **分叉（Fork）** 或 **重组（Reorg）** 难以完全避免。Hyperliquid 的引擎通过 **多层保护机制** 降低影响：

快速检测与回滚

- 监听共识节点状态，实时检测链重组信号
- 将未确认交易重放至最新链状态

状态快照

- 每个确认区块生成状态快照

- 在分叉时快速恢复到最近一致的状态

资金安全

- 在回滚过程中，未最终确认的撮合不影响资金余额
- 保证资金安全与一致性

2.6 批撮合的确定性约束

Hyperliquid 的批撮合机制要求 **同一批次交易** 在所有节点执行结果完全一致，避免状态分叉。公式如下：

$$\text{State}_{t+1} = f(\text{State}_t, \{\text{Orders}_i^{(t)}\})$$

其中：

- State_t：上一时刻链上状态
- $\{\text{Orders}_i^{(t)}\}$：该批次内的订单集合
- f：确定性撮合函数，保证相同输入 → 相同输出

这种严格的确定性约束，是链上可验证性的基础。

2.7 性能特征与优化总结

环节	延迟优化策略	效果
Sequencer 排序	微批次窗口 + 内存队列	减少抢跑，控制 <100ms 排队延迟
共识出块	BFT 快速出块	平均 ~1s 区块时间
状态写入	增量存储 + Merkle 压缩	TPS 提升至 5,000+
事务一致性	幂等执行 + 原子提交	避免脏状态和不一致

---

3. 订单簿与撮合引擎

Hyperliquid 的订单簿和撮合引擎（Matching Engine）是整个平台的“心脏”，决定了价格发现的速度、交易执行的公平性以及整体吞吐能力。在链上环境下实现接近中心化交易所（CEX）的性能体验，需要在数据结构、撮合逻辑和系统优化上进行深度定制。本章节将从 **订单数据结构、撮合流程、公平性机制 到 性能工程，**全面解析 Hyperliquid 的核心引擎。

3.1 订单表示：PriceLevel 与 OrderNode

3.1.1 PriceLevel 结构

订单簿的核心是 **价格层（Price Level）**，每个价格层记录该价格下的挂单链表。

Code block

```
struct PriceLevel {
    price        // 当前价格
    volume       // 总挂单量
    headOrder    // 挂单链表的头指针
    tailOrder    // 挂单链表的尾指针
```

```
}
```

每个 PriceLevel 通过链表结构串联挂单，支持快速遍历与撮合。

3.1.2 OrderNode 结构

链表中的每个节点（OrderNode）代表一个具体订单：

Code block

```
struct OrderNode {
    orderId
    traderAddress
    side            // 买 (Bid) 或 卖 (Ask)
    quantity
    filledQuantity
    timestamp
    nextOrder
    prevOrder
}
```

这种结构使得 **FIFO（时间优先）** 与 **Price-Time 优先** 的撮合机制能够高效实现。

3.1.3 Level2 订单簿示意

Code block

```
BID SIDE                          ASK SIDE
-------------------------------------------
Price  | Volume              Price  | Volume
-------------------------------------------
100.5  |  50               100.6  |   30
100.4  |  80               100.7  |   25
100.3  | 120               100.8  |   40
```

- 买单按价格从高到低排列
- 卖单按价格从低到高排列
- 同一价格下按时间顺序（FIFO）执行

3.2 订单类型与路由

Hyperliquid 支持丰富的订单类型与时间指令（TIF），满足从普通用户到专业做市商的不同需求。

3.2.1 订单类型

订单类型	描述	使用场景
Limit	限价单	做市、挂单策略
Market	市价单	快速成交、追单
PostOnly	只挂单	确保订单进入簿内不立即成交
ReduceOnly	仅减仓	用于部分平仓或止盈止损

3.2.2 TIF（Time in Force）支持矩阵

指令	含义	适用策略
GTC (Good Till Cancel)	挂单直到撤销	普通限价挂单
IOC (Immediate Or Cancel)	立即成交，否则取消未成交部分	高频策略、止损单
FOK (Fill Or Kill)	全部成交，否则全部取消	大额限价订单

3.2.3 触发单

- **Stop Order（止损单）**
- 当市场价格触发条件价时，自动转化为市价单执行。
- **TP Order（止盈单）**
- 当达到目标价时，自动市价平仓，保护利润。

3.2.4 路由逻辑

撮合引擎根据以下优先级处理路由：

1. ReduceOnly 指令优先
2. Market 单次于当前最优价撮合
3. Limit 单根据价格优先/时间优先排队
4. PostOnly 单检测是否会立即成交，若会则拒绝或调整

3.3 撮合循环

撮合引擎的核心循环如下：

3.3.1 撮合逻辑伪码

Code block

```
function match(order):
    if order.side == BUY:
        while order.quantity > 0 and bestAsk <= order.price:
            tradeQty = min(order.quantity, bestAsk.volume)
            executeTrade(order, bestAsk, tradeQty)
            updateOrderbook(bestAsk, tradeQty)
    else:
        while order.quantity > 0 and bestBid >= order.price:
            tradeQty = min(order.quantity, bestBid.volume)
            executeTrade(order, bestBid, tradeQty)
            updateOrderbook(bestBid, tradeQty)

    if order.quantity > 0:
        addToOrderbook(order)
```

3.3.2 撮合过程

价格穿越（Price Crossing）

- 如果买单价格 ≥ 当前卖一价（Best Ask），立即成交
- 如果卖单价格 ≤ 当前买一价（Best Bid），立即成交

部分成交（Partial Fill）

- 当挂单量不足时，部分成交，剩余挂单继续排队

剩余委托回簿

- 未成交部分挂单进入订单簿，等待后续撮合

原子性（Atomicity）

- 撮合和账户更新在同一事务内执行，确保一致性

3.4 公平性与反抢跑

3.4.1 连续双边竞价 vs 频拍（Frequent Batch Auction）

模式	特点	优劣分析
连续双边竞价	实时撮合、低延迟	高频用户有优势，易产生抢跑
频拍（FBA）	在微批窗口（50-100ms）内统一撮合	提升公平性，但增加轻微延迟

Hyperliquid 在 **默认连续竞价** 下运行，同时引入 **微批撮合** 选项，平衡公平性与性能。

3.4.2 取消 / 改价的节流

- 限制取消/改价的频率，避免订单簿刷单攻击
- 为高频交易增加冷却时间（如 50ms 内不得重复撤单）

3.5 性能工程

要在链上实现接近中心化交易所的体验，Hyperliquid 对撮合引擎进行了系统级优化。

3.5.1 批量撮合（Batch Matching）

- 将微秒级订单聚合成批次处理
- 提升吞吐量并减少锁竞争

3.5.2 快照组播（Snapshot Broadcast）

- 将订单簿快照实时组播到所有节点
- 降低状态同步延迟
- 支持轻量客户端快速同步

3.5.3 高效内存结构

结构	用途	特点
跳表（Skip List）	高效维护有序价格链	O(log n) 插入、删除
堆（Heap）	优先队列	快速定位最优价位
环形缓冲区（Ring Buffer）	撮合队列缓存	低延迟、无锁设计

3.5.4 p50 / p99 延迟预算

- **p50 延迟 < 100ms**
- **p99 延迟 < 300ms**
- 通过无锁结构、内存池优化和批次撮合实现低尾延迟

3.5.5 热点锁分解

- 将全局订单簿锁拆分为多层局部锁：
  - **价格级锁**：每个 PriceLevel 单独加锁
  - **账户级锁**：更新余额或仓位时局部加锁
- 提高并发性能，避免热点竞争

3.6 小结

Hyperliquid 的订单簿与撮合引擎在 **数据结构、撮合逻辑、公平性机制** 和 **系统优化** 四个方面实现了突破：

- 高效的 **PriceLevel + OrderNode** 结构，支撑多维度的撮合策略
- 支持多样化订单类型和灵活的时间策略（TIF）
- 在公平性和性能之间通过 **连续竞价 + 微批策略** 找到平衡
- 系统优化让链上性能接近 CEX，为高频和机构交易提供了可能。

这一整套设计，使 Hyperliquid 真正成为链上 **高性能 CLOB 基础设施** 的代表。

---

4. 永续合约机制（Perps）与资金费率

Hyperliquid（HL）的永续合约（Perpetual Futures, 简称 **Perps**）是其生态的核心产品模块，面向高频交易者、量化团队以及机构用户，提供链上可验证、低延迟的永续交易体验。其机制融合了 **指数价 / 标记价双轨定价**、**资金费率驱动的市场均衡机制**、**高精度的 PnL 计算** 和 **统一保证金系统**，确保了交易的公平性、稳定性和高效性。

本节从价格逻辑、资金费率、盈亏（PnL）记账、保证金体系以及合约参数化五个维度，深入解析 Hyperliquid 的永续合约技术设计。

4.1 指数价 / 标记价的定义与使用边界

永续合约的价格体系主要由 **指数价（Index Price）** 和 **标记价（Mark Price）** 两个核心概念构成，它们在交易撮合、PnL 结算、风险控制等环节扮演不同角色。

（1）指数价（Index Price）

指数价 P_{index} 是由多个权威数据源聚合生成的市场参考价，反映标的资产的 **全球加权现货价格**。

来源与计算：

- 数据源：Pyth、Chainlink、Kaiko 等多方预言机
- 算法：加权中位数（Weighted Median）或加权 TWAP（Time-Weighted Average Price）
- 异常剔除：移除极端价格点，避免预言机波动影响

主要用途：

- 资金费率计算
- 风险参数校验（如破产价、维持保证金）
- 自动清算与 ADL（自动减仓）的触发基准

（2）标记价（Mark Price）

标记价 P_{mark} 是用于计算未实现盈亏和清算逻辑的动态价格。

计算公式：

$$P_{mark} = P_{index} + \alpha \cdot (P_{perp} - P_{index})$$

其中：

- P_{perp}：当前合约交易价格
- α：调节因子（通常 < 1）

主要用途：

- 动态计算未实现盈亏（Unrealized PnL）
- 控制清算触发点，避免因瞬时波动导致无谓清算

（3）使用边界对比

维度	指数价 (Index)	标记价 (Mark)
价格基准	全球市场参考价	指数价加微调偏移
用途	资金费率、破产价、风险参数	PnL、清算判定
波动敏感度	较低	较高
稳定性	高	中等

4.2 资金费率（Funding）

资金费率（Funding Rate）是永续合约价格稳定的关键机制，用于将 **合约价格** P_{perp} 锚定至 **指数价** P_{index}。

（1）资金费率公式

资金费率的计算遵循以下公式：

$$premium = \text{TWAP}\left(\frac{P_{perp} - P_{index}}{P_{index}}\right)$$

$$funding = \text{clamp}(k \cdot premium + \Delta r, \pm f_{max})$$

其中：

- premium：价格溢价（正值表示多头溢价，负值表示空头溢价）
- k：调节系数
- \Delta r：基准利差，通常为 0 或微小正值
- f_{max}：资金费率上限，防止极端行情下失控

（2）资金费率结算周期

- 标准周期：**每小时一次（1h）** 或 **每 8 小时一次（8h）**
- 高波动模式：在剧烈行情下缩短至 30 分钟或动态调整周期
- 每个周期计算 TWAP，平滑短期波动，避免频繁资金转移

（3）钳制与异常期策略

在异常行情或预言机失效时，资金费率会被 **钳制（Clamp）** 到安全范围内：

- 限制资金费率最大值，如 ±0.75% / 周期
- 暂停资金结算，直至市场恢复正常
- 增加保证金要求，降低风险

（4）资金费率结算时序

资金费率的结算流程：

1. 系统根据 TWAP 差值计算 Funding Rate
2. 每周期对所有持仓用户计算应付或应收资金费
3. 在账户余额中直接扣减或增加

4.3 PnL 结算与会计

PnL（Profit and Loss，盈亏）的精确计算，是保证市场公平与透明的基础。

（1）PnL 公式

PnL 的核心公式如下：

$$\text{PnL} = \pm(P_t - P_{entry}) \cdot \text{contract\_size}$$

其中：

- P_t：当前价格
- P_{entry}：开仓价格
- contract_size：合约张数 × 合约乘数
- 正号 +：多头仓位
- 负号 -：空头仓位

（2）未实现盈亏（Unrealized PnL）

基于标记价 P_{mark} 实时计算，用于：

- 保证金占用率计算
- 清算触发条件监测

（3）已实现盈亏（Realized PnL）

基于平仓价格与开仓价格的差值，在仓位关闭后结算。

（4）盈亏记账模式

类型	计价基准	结算时间
未实现盈亏	标记价	实时计算
已实现盈亏	成交价	平仓时确认
资金费率盈亏	资金费率	每周期结算

4.4 统一保证金与组合保证金

Hyperliquid 支持两种保证金模式：

（1）统一保证金（Cross Margin）

- 所有仓位共享一个账户权益（Equity）。
- 有利于资金利用率最大化，但风险也集中。
- 适合高频量化或低杠杆多策略组合。

（2）组合保证金（Portfolio Margin）

- 基于多品种相关性计算净风险。
- 计算公式：

$$Risk = \sqrt{\sum_i Position_i^2 + 2\sum_{i<j} \rho_{ij} Position_i Position_j}$$

其中 \rho_{ij} 为品种间的相关性系数。

- 优点：显著降低高相关资产的保证金占用。
- 缺点：实现复杂，需要实时风险计算引擎。

（3）风险监控与触发

- 实时计算 Margin Ratio（保证金比率）
- 当账户权益低于维持保证金（MM）阈值时，触发强制平仓

4.5 永续参数化

合约的参数化是保证市场流动性和稳定性的基础。

（1）合约乘数（Contract Multiplier）

定义每张合约对应的标的资产数量。例如：

- BTC 合约：1 张 = 0.001 BTC
- ETH 合约：1 张 = 0.01 ETH

（2）最小变动价位（Tick Size）

每个合约的最小报价精度，例如：

- BTC：0.5 USDT

- ETH：0.05 USDT

（3）最小下单量

控制合约的最小下单单位，避免小额交易对系统造成压力。

（4）限价带（Price Band）

设置动态限价范围，防止用户提交远离市场的订单，提升撮合稳定性。

（5）核心市场参数模板

参数	BTC	ETH	SOL
合约乘数	0.001	0.01	1
Tick Size	0.5	0.05	0.001
最小下单量	1 张	10 张	50 张
初始保证金 (IM)	5%	5%	10%
维持保证金 (MM)	3%	3%	6%

4.6 小结

Hyperliquid 的永续合约机制在以下方面具备突出优势：

- **精准的三价模型**：通过指数价和标记价分离，兼顾价格稳定性与风险响应速度。
- **资金费率驱动均衡**：动态调节机制确保合约价格锚定现货市场。
- **高精度的盈亏会计**：原子化账本记录，支持高频策略的实时计算需求。
- **先进的保证金体系**：统一保证金与组合保证金的灵活支持，使资金利用效率和风险控制达到平衡。
- **参数化灵活**：为多资产、多策略交易场景提供一致且透明的市场规则。

这一整套机制，使得 Hyperliquid 的永续合约既具备链上透明性，又能满足专业机构和高频策略团队的苛刻性能与风险管理需求。

---

5. 风险引擎与清算体系

在链上高性能永续合约平台中，**风险引擎**（Risk Engine）是整个系统的安全基石，而 **清算体系**（Liquidation System）则是风险控制策略的执行落地。Hyperliquid（HL）通过精细化的保证金管理、实时账户风险计算、自动化清算流程、透明的保险基金管理和 ADL（Auto-Deleveraging，自动减仓）机制，构建了一套能在极端市场环境中保持稳健运行的完整风控架构。

5.1 保证金体系

5.1.1 初始保证金（IM）与维持保证金（MM）阶梯

Hyperliquid 的保证金采用 **分层阶梯化结构**，根据持仓的名义规模动态调整 IM（Initial Margin，初始保证金）和 MM（Maintenance Margin，维持保证金）。

名义仓位（USD）	初始保证金 (IM)	维持保证金 (MM)
0 – 100,000	5%	2.50%
100,001 – 1,000,000	7.50%	3.50%
1,000,001+	10%	5%

特点：

- **动态风险缓释**：仓位越大，保证金比例越高。
- **减少尾部风险**：避免高杠杆集中于少数大户，提升系统韧性。
- **灵活调整**：支持参数热更新，能快速响应波动市场。

5.1.2 多资产抵押与 Haircut

在 **统一保证金模式** 下，用户可以使用多种资产（如 USDC、USDT、ETH）作为抵押品，但不同资产会有 **Haircut（折扣因子）**：

资产	Haircut（折扣率）	说明
USDC / USDT	0%	稳定币，无折扣
ETH	10%	价格波动较大
BTC	12%	波动风险更高
其他长尾资产	30%+	仅做补充抵押

折扣计算公式：

$$\text{Collateral Value}_j = \text{Balance}_j \times \text{Price}_j \times (1 - h_j)$$

这种机制确保 **抵押价值更真实、风险可控**，即使标的资产波动剧烈，也不会迅速侵蚀系统稳定性。

5.2 账户风险计算

风险引擎实时计算每个账户的 **权益（Equity）** 和 **保证金比率（Margin Ratio）**。

5.2.1 核心公式

账户权益：

$$\text{Equity} = \text{Cash} + \sum_i \text{UPnL}_i - \sum_j \text{Haircut}_j$$

保证金比率：

$$\text{Margin Ratio} = \frac{\text{Equity}}{\text{Maintenance Margin Requirement}}$$

变量解释：

- **Cash**：账户可用现金余额
- **UPnL**：未实现盈亏（Unrealized Profit and Loss）
- **Haircut**：折扣后的抵押品价值
- **Maintenance Margin Requirement**：维持保证金需求

5.2.2 风险分层

风险等级	保证金比率（MR）	系统动作
安全	> 150%	正常交易
警戒	120% – 150%	限制加仓
危险	100% – 120%	自动通知用户，触发部分减仓预警
清算	⩽ 100%	启动强制清算流程

5.3 清算流程

Hyperliquid 的清算分为 **部分清算** 与 **全额清算**，并支持链上实时执行。

5.3.1 清算触发条件

当：

$$\text{Equity} \leq \text{Maintenance Margin Requirement}$$

触发自动清算。

5.3.2 清算执行路径

触发信号

风险引擎检测到账户权益不足，生成清算任务。

部分清算

- 先以最小数量分批平仓
- 逐步提升账户保证金比率至安全区间

全额清算

- 若部分清算无法满足安全阈值，平掉所有仓位

订单簿执行

- 清算指令作为市价单进入订单簿
- 以最佳可得价格成交

失败回退与重试

- 如果因流动性不足导致成交失败，系统重试或进入“拍卖模式”

5.3.3 破产价与穿仓处理

- **破产价（Bankruptcy Price）**
    - 当仓位价值归零时的价格水平
- **穿仓（Negative Equity）**
    - 当市场波动剧烈导致清算无法覆盖亏损时，触发保险基金或 ADL 机制填补缺口

5.3.4 清算折价

清算仓位通常会给予 **折价（Discount）**：

仓位规模	折价率
⩽ 100,000 USD	0.25%
100,001–1,000,000 USD	0.50%
⩾ 1,000,001 USD	1.00%

折价确保快速成交并激励做市商参与清算。

5.4 保险基金与 ADL（自动减仓）

5.4.1 保险基金

注资与累积

- 来源：交易手续费、清算收益部分注入
- 用途：覆盖极端行情中穿仓损失
- 公开透明：基金余额、流动方向实时链上可查

5.4.2 ADL（Auto-Deleveraging 自动减仓）

当 **保险基金不足** 或 **市场极度单边化** 时，系统启用 ADL。

触发条件：

- 市场价格快速跳空
- OI（未平仓量）极端偏斜
- 清算无法快速完成

执行逻辑：

- 根据杠杆率和盈利排名对用户仓位进行分级
- 按优先级递减自动减仓

排名优先级	标准
1	高杠杆 + 高盈利
2	高杠杆 + 中盈利
3	低杠杆 + 中盈利
4	低杠杆 + 低盈利

5.5 极端波动与熔断机制

5.5.1 价格带保护

- 限制单笔订单价格偏离标记价过远
- 防止因输入错误或操纵行为导致异常成交

5.5.2 交易节流

- 高频风控策略
- 限制异常短时间内的大规模撤单和下单请求

5.5.3 只减仓模式（Reduce-Only Mode）

- 在市场剧烈波动时，平台强制进入只减仓状态
- 禁止新增仓位，专注于降低系统整体风险

5.5.4 风险参数在线治理

- 风控参数（IM、MM、折价率等）支持热更新
- 社区或治理委员会可快速调整参数以应对特殊行情

5.6 小结

Hyperliquid 的风险引擎和清算体系具备以下优势：

- **动态化保证金管理**：分层 IM/MM 与多资产抵押 Haircut 提供灵活与稳健的平衡。
- **实时账户风控**：保证高频场景下账户风险的精确监测与响应。
- **稳健的清算执行**：原子化撮合与快速回滚机制，降低尾部风险。
- **透明保险基金 + ADL**：链上可验证的风险缓冲，确保极端行情下的安全性。
- **应对极端波动的防御策略**：从价格带保护到熔断和 Reduce-Only 模式，全面防御系统性风险。

这一整套设计，使 Hyperliquid 在保证链上透明性与可验证性的同时，能够承载机构级交易者在复杂、快速波动的市场中稳定运行。

---

6. 预言机与价格基础设施

在去中心化衍生品市场中，**预言机（Oracle）** 是系统稳定运行的生命线，承担着为 **价格发现、风险管理、清算机制** 提供实时、准确、抗操纵的市场价格数据。Hyperliquid（HL）通过多源聚合、智能异常检测、时序质量控制、失效兜底及风控约束，构建了一套高性能的 **价格基础设施**，确保链上价格机制在极端行情下依然稳健、安全。

6.1 多源聚合机制

Hyperliquid 的价格体系核心在于 **多源聚合**，通过集成 **Pyth、Chainlink 以及自研聚合引擎**，消除单一预言机可能带来的偏差和攻击风险。

6.1.1 数据源组成

- **Pyth Network**
  - 提供高速链上价格更新，适合高频交易需求。
  - 数据覆盖率广，尤其适合主流加密资产。
- **Chainlink**
  - 稳定性强，历史表现稳健。
  - 适合稳定币、RWA 等低频更新标的。
- **自研聚合引擎**
  - 聚合来自 CEX（Binance、OKX、Bybit）和其他 DEX 的价格。
  - 提供灵活的权重调节和冗余机制。

6.1.2 聚合逻辑

价格聚合使用 **中位数 + 加权模型**：

$$P_{agg} = \text{Median}\left(w_i \cdot P_i\right)$$

其中：

- P_i：第 i 个数据源的报价
- w_i：权重（根据数据源可靠性与延迟调整）

特点：

- **鲁棒性强**：中位数防御极端异常点
- **动态加权**：高信誉数据源（如低延迟、高可靠性）占更高权重

6.1.3 信誉模型

HL 内置 **信誉评分机制**，综合以下指标动态调整数据源权重：

- 数据延迟（Latency）
- 价格准确率（Accuracy）
- 报价频率（Heartbeat Consistency）
- 异常率（Deviation Incidents）

6.2 时序质量控制

价格数据不仅要准确，还需要具备高 **时序质量**，以支持低延迟的链上撮合与风险管理。

6.2.1 心跳机制（Heartbeat）

- 每个价格源按照固定间隔推送心跳信号
- 若超过心跳周期未更新，立即降级该价格源权重

6.2.2 陈旧度检测（Staleness Check）

定义 **陈旧阈值**（如 3 秒）：

当价格更新时间超过阈值，触发以下策略：

- 剔除陈旧数据
- 启用备用源
- 提高风险参数（如提高 IM、触发 Reduce-Only）

6.2.3 时间漂移（Time Drift）

- 比较链上时间戳与价格源时间戳
- 超过漂移阈值（如 ±1 秒）直接丢弃异常数据

6.3 异常检测与过滤

为了防止价格操纵或预言机异常，HL 采用多层异常检测机制。

6.3.1 偏差熔断（Deviation Threshold）

设定 **最大偏离阈值**：

$$| P_{new} - P_{median} | > \delta \cdot P_{median}$$

如果价格跳变超过阈值（如 ±0.5%），自动熔断并忽略该价格。

6.3.2 离群点检测

基于统计学方法检测异常价格：

- **Z-Score**：
- $z = \frac{P_i - \mu}{\sigma}$
- 超过阈值（如 $|z|>3$）即视为离群点。
- **IQR（四分位距）**：
- $\text{Outlier} = P_i > Q3 + 1.5 \cdot IQR \quad 或 \quad P_i < Q1 - 1.5 \cdot IQR$

6.4 失效兜底机制

在极端情况下，预言机可能发生同步失效，HL 内置了多层 **兜底保护机制**：

6.4.1 失效检测

当所有主要数据源同时失效：

- 无更新信号
- 或价格波动超出预设安全区间

系统立即进入失效模式。

6.4.2 兜底逻辑

1. 使用 **最后一次有效价格** P_{last}
2. 启动更严格的风险参数：
    - 提高 **IM（初始保证金）** 要求
    - 强制 **Reduce-Only** 模式
    - 禁止高杠杆新开仓

6.5 指数价 / 标记价差的风控约束

Hyperliquid 设置 **价格约束区间**，防止异常行情或操纵行为影响清算和撮合。

6.5.1 最大偏离限制

- 限制指数价 P_{index} 与标记价 P_{mark} 的偏差：
- | P_{mark} - P_{index} | \leq \delta
- 常见参数范围为 ±0.25% ~ ±1%。

6.5.2 成交带限制（Price Banding）

- 限制订单的成交价格需在当前指数价一定范围内
- 防止远离市场价格的“钓鱼单”或撮合异常

6.6 TWAP 与 Median 窗口公式

6.6.1 TWAP（时间加权平均价格）

$$P_{TWAP} = \frac{\sum_{i=1}^{n} P_i \cdot \Delta t_i}{\sum_{i=1}^{n} \Delta t_i}$$

6.6.2 Median（中位数）

对于排序后的价格数组：

$$P_{median} = \begin{cases} P_{\frac{n+1}{2}}, & n \text{ 为奇数} \\ \frac{P_{\frac{n}{2}} + P_{\frac{n}{2}+1}}{2}, & n \text{ 为偶数} \end{cases}$$

6.7 小结

Hyperliquid 的预言机和价格基础设施具有以下核心优势：

- **多源聚合**：通过 Pyth、Chainlink、自研聚合实现价格多样性和抗操纵能力。
- **高时序质量**：心跳检测、陈旧度校验与时间漂移控制确保实时性和可靠性。
- **智能异常检测**：偏差熔断、统计离群点过滤，防止价格操纵攻击。
- **多层兜底机制**：在极端行情或预言机失效时，保证交易和风控稳定运行。
- **风控约束**：通过最大偏离限制和成交带机制，强化链上撮合安全性。

这套价格基础设施既满足了 **高频交易的低延迟需求**，又保障了 **链上价格透明性与安全性**，为 Hyperliquid 的衍生品生态提供坚实支撑。

---

7. 账户模型与结算记账

Hyperliquid（HL）的账户与结算架构设计，旨在兼顾 **专业机构级交易的灵活性** 和 **链上协议的透明性**。通过 **统一保证金账户（Cross Margin Account）**、**子账户体系（Subaccounts）**、**多抵押品管理** 以及 **高效的资金结算与费用体系**，Hyperliquid 构建了一套支持高频、跨市场、多资产交易的完整资金管理框架。

7.1 统一保证金账户与子账户体系

7.1.1 统一保证金账户（Cross Margin Account）

Hyperliquid 采用 **统一保证金模型（Cross Margin）**，即所有资产和仓位共享一个总权益池，用于抵押、交易和结算。

特点：

- **高资金效率**

  单一保证金池覆盖多品种交易，避免资金碎片化。
- **风险共享**

  盈利仓位的浮动收益可以弥补亏损仓位的保证金缺口。
- **动态杠杆调节**

  根据实时风险参数，系统可调整杠杆使用率。

适用场景

- 高频量化交易
- 多资产对冲策略
- 机构化大资金组合管理

7.1.2 子账户（Subaccount）

为满足 **多策略隔离** 或 **团队化管理** 的需求，Hyperliquid 支持 **子账户机制**：

- 每个子账户拥有独立的仓位与风险参数。
- 支持 **统一资金池下的风险隔离**。
- 子账户可绑定不同策略或 API Key，方便团队化风控。

示例：

账户类型	用途	特点
主账户	资金主控	决定资金总额度、授权策略
子账户 1	高频做市	高速 API、低延迟
子账户 2	期现套利	风险隔离，低杠杆
子账户 3	CTA 策略	中长周期交易

7.2 多抵押品结算

7.2.1 统一计价单位（Numéraire）

系统以 **稳定币（USDC 或 USDT）** 作为统一计价单位（Numéraire），方便计算账户权益、保证金及 PnL。

优势：

- 简化跨资产计算
- 保证资产计价稳定，降低波动性风险

7.2.2 跨币折算与折扣

当用户以多种加密资产作为抵押品时，系统通过 **实时折算 + 折扣（Haircut）** 计算有效抵押价值：

$$CollateralValue = \sum_i \big(Q_i \times P_i \times (1 - Haircut_i) \big)$$

折扣示例：

抵押资产	折扣率（Haircut）
USDC / USDT	0%
ETH	10%
BTC	12%
波动性资产	⩾ 30%

风控逻辑：

- 折扣率越高，风险抵御能力越强。
- 避免因高波动资产价格闪崩导致的穿仓风险。

7.3 资金划转、授权与风控额度

Hyperliquid 的账户体系支持灵活的资金划转与权限控制，特别适合多策略、多账户的交易团队。

7.3.1 资金划转

- 支持主账户与子账户之间实时划转
- 所有划转操作链上可追溯，具备透明性

7.3.2 授权管理

- 针对每个子账户可设置 **API Key 权限**
    - 下单权限
    - 划转权限
    - 读取数据权限

7.3.3 风控额度

- **按账户限制**：每个账户的最大仓位、最大杠杆可配置
- **按产品限制**：针对单一市场（如 BTC-PERP）设置独立风险上限

7.4 费用与返佣体系

Hyperliquid 采用 **Maker / Taker 双向费率** 机制，并结合 **阶梯式费率折扣** 和 **返佣计划**，激励流动性提供者和高频策略团队。

7.4.1 基础费率

角色	现货交易	永续合约
Maker	-0.005% （返佣）	-0.005% （返佣）
Taker	0.05%	0.05%

说明：

- Maker 享受挂单返佣，鼓励做市行为
- Taker 需支付较高手续费，抑制无序冲击

7.4.2 阶梯费率

根据 30 天滚动交易量（USD 等值）提供阶梯折扣：

交易量区间 (30 天)	Maker 费率	Taker 费率
0 – 1M	-0.01%	0.05%
1M – 10M	-0.01%	0.05%
10M – 50M	-0.01%	0.04%
50M+	-0.01%	0.04%

7.4.3 特殊费用

- **撮合附加费**：高负载期间可能临时提高撮合费用
- **清算附加费**：清算成交额的 0.5%，进入保险基金
- **OI 附加费**：未平仓量（Open Interest）极高时，对新增仓位加收 0.01%–0.03%

7.4.4 返佣计划

- 返佣周期：每日计算，每周发放
- 推荐计划：通过邀请链条分配一定比例返佣，鼓励生态扩展

7.5 资金结算流程

Hyperliquid 的结算引擎确保 **链上透明性** 与 **高效结算** 并行，覆盖 **入金、交易、结算、出金** 的全流程。

7.5.1 流程图

资金流转路径

Code block

```
[入金 Deposit]
      ↓
```

```
[账户权益计算]
      ↓
[交易撮合 Execution]
      ↓
[实时 PnL 结算]
      ↓
[费用扣减 + 资金划转]
      ↓
[出金 Withdraw]
```

7.5.2 过程解析

- **入金（Deposit）**

  用户通过链上操作将资金存入主账户或子账户。
- **交易（Execution）**

  订单成交后，PnL 和资金占用实时更新。
- **结算（Settlement）**

  统一保证金模式下，盈亏以稳定币计价，原子化记账，避免延迟和错配。
- **出金（Withdraw）**

  通过链上验证和冷钱包管理，确保资金安全。

7.6 小结

Hyperliquid 的账户模型与结算记账机制具备以下优势：

- **统一保证金账户** 提高资金使用效率，适合高频策略和机构用户。
- **子账户隔离** 满足多策略团队化交易的需求。
- **多抵押品支持** 和 **折扣管理** 有效防范价格剧烈波动风险。
- **灵活的划转与风控额度** 提供交易自由度的同时保障安全。

- **透明的费用和返佣体系** 激励做市商和量化团队积极参与。
- **链上实时结算** 确保交易透明与资金安全，降低对手方风险。

这种设计不仅满足了高频和大资金交易的复杂需求，也保持了链上金融协议的公开性和可验证性，为 Hyperliquid 的生态扩展提供了坚实的基础。

---

8. MEV、排序公平性与抗操纵

在链上高性能 CLOB（Central Limit Order Book，中心化限价订单簿）架构下，**MEV（Miner/Maximal Extractable Value，最大可提取价值）** 是交易公平性和安全性必须应对的核心挑战。Hyperliquid（HL）通过 **微批次撮合、撮合层速率限制、链上订单加密机制** 及 **价格保护策略** 等多层防御，构建了系统化的 **抗操纵体系**，确保高频交易和机构策略的执行安全性。

8.1 链上 CLOB 的 MEV 攻击面

CLOB 模式的优势在于实时的价格发现和深度撮合，但也使其成为 MEV 攻击的高发区。以下是 HL 在链上交易过程中面临的典型攻击面：

8.1.1 抢跑（Front-running）

- 攻击者通过监听 Mempool，优先提交相同或反向订单，抢占价格优势。
- 场景：
    - 用户挂出大额买单，攻击者提前吃单或挂高价抢占队列。

8.1.2 插队（Reordering）

- 节点或验证者故意调整交易顺序，将自身或关联地址的订单提前执行。

8.1.3 尾随（Back-running）

- 当某笔大额订单触发市场波动时，攻击者快速跟随下单，从价格惯性中套利。

8.1.4 Sandwich 攻击

- 攻击者先挂单推动价格，然后在目标订单执行后反向平仓，利用价格波动套利。

8.2 抗 MEV 策略

Hyperliquid 的抗 MEV 策略涵盖从 **交易提交前 → 排序与撮合中 → 成交后状态更新** 的全链条。

8.2.1 批次撮合（FBA：Frequent Batch Auction）

采用 **微批次时间窗**（50ms–200ms）机制，将该时间窗内的订单批量撮合，消除抢跑优势。

机制：

- 收集窗口内的所有订单，按价格优先 / 时间优先排序
- 在批次结束后统一撮合

优点：

- 有效降低抢跑攻击的成功率
- 在可控延迟范围内平衡公平性与性能

伪代码示例：

Code block

```
for each batch_window:
    orders = collect_orders(time_window=100ms)
    sort_orders_by(price, timestamp)
    match_orders_in_batch(orders)
```

8.2.2 取消 / 改价速率限制

为防止 **刷单操纵深度** 或 **诱导策略**，系统设置 **撤单和改价冷却时间**：

- 每账户每秒可撤单次数上限，例如 20 次 / 秒
- 改价冷却时间，例如 50ms 内不得连续修改同一订单
- 高频策略需通过特殊权限验证 API Key，确保公平性

8.2.3 订单加密与 Commit-Reveal

Hyperliquid 支持 **订单提交加密机制** 的探索，基于 **Commit-Reveal** 双阶段：

1. **Commit 阶段**：提交哈希（订单内容 + 随机盐）
2. **Reveal 阶段**：在下一批撮合周期揭示订单详情

优点：

- 完全消除抢跑风险
- 增强撮合公平性

现实挑战：

- 增加交易延迟
- 用户体验受限，尤其是高频策略下

因此，在主市场运行中，HL 更倾向于 **微批次撮合 + API 节流** 的轻量化方案。

8.3 预言机操纵与价格保护

除了交易顺序层面的 MEV，价格相关的操纵风险同样是关键威胁。

8.3.1 价带保护（Price Band）

- 限制挂单价格偏离 **指数价（Index Price）** 的最大范围，例如 ±1%。
- 防止恶意刷单操纵市场深度或价格。

8.3.2 最大滑点限制（Max Slippage）

- 对所有市价单设置最大可承受滑点（如 0.5%）。
- 防止大额订单成交价格被恶意拉高或压低。

8.3.3 限价保护（Limit Protection）

- 对挂单撮合执行二次验证：
    - 若市场波动超过预设阈值，订单自动失效或延迟撮合。
    - 清算触发单严格绑定指数价，以防异常波动导致无谓清算。

8.4 MEV 防护分层体系

Hyperliquid 的 MEV 抗性不是单一机制，而是 **多层控制点的组合**。

8.4.1 上链前（Pre-Chain）

- API 网关速率限制
- SDK 提交签名校验
- 提前过滤无效或恶意订单

8.4.2 排序与撮合中（In-Chain）

- 微批次撮合（FBA）
- 改价 / 撤单速率控制
- 时间优先 + 哈希随机性

8.4.3 成交后（Post-Chain）

- 快照同步，保证结果一致性

- 异常监控与惩罚机制
- 异常模式下自动进入只减仓（Reduce-Only Mode）

8.5 设计权衡

Hyperliquid 在抗 MEV 的策略设计上遵循 **公平性、性能、可用性三者平衡** 原则：

策略	优点	缺点
微批次撮合	延迟可控、实现简单	抢跑完全消除效果有限
Commit-Reveal	完全抵御抢跑	增加延迟，降低高频策略体验
速率限制	阻断刷单操纵	高频交易需要 API 白名单
价带保护	防止价格操纵	波动剧烈行情下可能影响流动性

8.6 小结

Hyperliquid 在链上 CLOB 架构中构建了完整的 **抗 MEV 与排序公平性体系**：

- 通过 **微批次撮合** 和 **撤单冷却** 平衡低延迟和公平性。
- 结合 **订单加密机制** 和 **价格保护策略**，防御抢跑、插队和预言机操纵。
- 构建 **上链前 / 排序中 / 成交后** 的全链路多层控制体系，实现安全、透明且高效的撮合体验。

这种多层次的防御策略，让 Hyperliquid 在保持高性能的同时，确保市场的公平性和安全性，为高频交易、做市策略以及机构化参与提供了稳健的技术保障。

---

9. 安全工程与合规模块

Hyperliquid（HL）作为一条 **高性能链上 CLOB 永续交易平台**，其安全和合规体系是支撑整个生态稳定运行的基础。

在一个高频交易、复杂交互的链上系统中，安全风险与合规需求都远高于普通 AMM 或 Spot DEX，因此 HL 采用 **“合约安全 + 运行时防护 + 合规接口”** 的三层防御架构，辅以多维度监测和治理能力，确保平台的稳健运行。

9.1 智能合约安全

智能合约是链上系统的底座，HL 在设计上通过 **形式化验证、不可变性约束、多签治理** 等手段，将安全性固化在架构层。

9.1.1 形式化断言（Formal Verification）

- 在核心模块（撮合引擎、账户管理、结算系统）中引入 **形式化验证框架**（如 Certora、Foundry 的 formal assertion），用逻辑证明关键路径安全。
- 重点验证内容：
    - **状态一致性**：账户余额、仓位与账本哈希一致

- **原子性**：撮合与结算不可分割
- **风险约束**：权益不足时禁止下单与划转

9.1.2 不可变性约束

- 核心逻辑合约 **一经部署不可升级**，防止治理层恶意篡改。
- 可升级模块（如前端适配层）通过链上白名单控制权限，确保代码变更可审计。

9.1.3 临界函数多签

- 涉及关键权限的函数，均需 **多签确认（Multisig）**：
  - 预言机白名单管理
  - 风险参数修改
  - 保险基金调拨
- 多签门槛：3-of-5 或 4-of-7，确保即使部分密钥泄露也无法影响全局安全。

9.2 审计与测试体系

安全不仅依赖部署前的审计，还需要 **持续测试与实时监控**。

9.2.1 双审计（Dual Audit）

- 每次核心合约更新，必须经过 **两家独立安全审计机构** 完整审计。
- 审计范围：
  - 逻辑漏洞
  - 经济攻击（如闪电贷操纵、资金迁移漏洞）
  - 性能边界条件

9.2.2 持续模糊测试（Continuous Fuzzing）

- 部署后，核心合约与撮合引擎持续运行 **fuzzing 测试**。
- 随机生成交易路径并验证以下不变量（Invariants）：
  - 账户总权益守恒
  - 撮合价格不出现负数或无效价格
  - 保证金账户不出现非法穿仓

9.2.3 漏洞赏金（Bug Bounty）

- 常态化的白帽赏金计划，鼓励安全研究员提交漏洞。
- 奖励梯度：

- 严重漏洞：最高 500,000 USDC
- 中风险漏洞：50,000–100,000 USDC
- 低风险漏洞：5,000–10,000 USDC

9.3 运行时守护与防御

合约安全之外，运行时风险防控是 Hyperliquid 稳定性的第二层屏障。

9.3.1 风控守门人（Guardian）

- 运行时监控市场状态与链上交易，实时干预可疑行为：
  - 异常波动（如 ±20% 快速波动）
  - 账户集中度过高
  - 大额订单导致滑点剧烈增加

触发时，Guardian 模块可：

- 暂停撮合或指定市场
- 降低杠杆上限
- 强制 Reduce-Only 模式

9.3.2 参数热修

- 支持风控参数的链上热更新，例如：
  - IM/MM（初始/维持保证金比例）
  - 单市场最大 OI 限制
  - 单账户最大风险限额
- 所有变更均链上记录，保证透明性与可追溯性。

9.3.3 熔断开关（Circuit Breaker）

- 在极端情况下触发熔断：
  - 预言机价格异常失效
  - 撮合延迟大幅升高
  - 清算量超过安全阈值
- 熔断状态下仅允许 **平仓/减仓** 操作，保护市场稳定。

9.4 合规接口

在全球监管趋严的背景下，Hyperliquid 内置了多层 **合规控制与接口**，以满足多市场合规需求。

9.4.1 KYT / KYB

- **KYT（Know Your Transaction）**
  - 实时监控链上地址与交易行为，识别高风险资金流动。
  - 集成 Chainalysis、TRM 等服务。
- **KYB（Know Your Business）**
  - 针对机构用户验证注册信息、经营合规性。

9.4.2 黑白名单管理

- **黑名单**：受制裁地址或与非法活动相关的账户自动限制交易。
- **白名单**：合规机构用户可快速通过审批，享受更高额度与 API 权限。

9.4.3 域白名单

- 防止 API 钓鱼攻击，仅允许授权域名访问 API 网关。

9.4.4 可审计交易标签

- 为每笔交易生成 **链上审计标签**，记录：
  - 用户地址
  - 下单时间
  - 撮合路径
  - 成交价格与手续费
- 提供链上可追溯的完整历史记录。

9.5 关键不变量（Invariants）清单

为了确保系统的稳定性与正确性，Hyperliquid 定义了一系列 **关键不变量**，并在运行时持续校验：

模块	不变量	触发响应
撮合引擎	总成交额 = 买单成交额 = 卖单成交额	强制中止交易循环，触发报警
风险引擎	保证金 ≥ 最低维持保证金	限制新订单，触发部分清算
账户系统	总权益守恒（Equity Conservation）	冻结账户，人工复查
清算模块	保险基金余额 ≥ 预警阈值	注资或启动 ADL 机制

9.6 合规检查点

环节	检查点	触发动作
Pre-Trade	地址是否在黑名单；杠杆是否超过上限	拒绝下单 / 发出警告
In-Trade	交易路径是否异常；资金来源可追溯性	降低杠杆；人工风控介入
Post-Trade	异常盈亏或快速大额资金转移	延迟提款，风控复核

9.7 小结

Hyperliquid 的 **安全工程与合规模块** 通过 **三层架构 + 全流程监控**，实现了从底层合约到运行时再到合规治理的全链路安全防护：

- **智能合约层**：形式化验证、不可变性设计、多签机制确保基础安全。
- **运行时层**：Guardian 风控、参数热修与熔断机制确保极端行情下稳定运行。
- **合规层**：KYT/KYB、黑白名单与审计标签支持全球市场的合规接入。

这套安全与合规体系，为 Hyperliquid 在全球化市场中稳定扩展提供了坚实的技术与监管保障，也使其具备了接入机构资金和合规生态的能力。

---

10. 可观测性与运维（Observability & Operations）

高性能链上衍生品交易平台的稳定运行，离不开 **可观测性（Observability）** 和 **专业化运维体系（DevOps/SRE）** 的支持。Hyperliquid（HL）作为一个集撮合、清算、预言机、风控于一体的 **链上 CLOB 永续交易平台**，需要持续监控全链路的性能与健康状态，并具备快速响应故障与弹性扩容的能力。本节从 **指标体系、日志与追踪、告警机制、压测与容量规划** 四个方面，系统剖析 Hyperliquid 的可观测性与运维策略。

10.1 核心指标体系（Metrics）

可观测性的核心是指标监控，HL 构建了从业务层到系统层的多维指标体系。

10.1.1 性能指标

指标	说明	监控目标
撮合延迟（Matching Latency）	从订单提交到成交确认的时间	p50 < 100ms，p99 < 300ms
拒单率（Rejection Rate）	因余额不足、参数错误等被拒绝订单比例	< 0.1%
区块同步延迟（Block Sync Latency）	节点区块同步延迟	< 1s

10.1.2 风险与资金指标

指标	说明	监控目标
资金费率异常（Funding Anomalies）	实际资金费率 vs 理论值的偏差	偏差 < 0.05%
清算命中率（Liquidation Accuracy）	清算价格与理论破产价一致性	> 99.5%
保险基金余额曲线	监控保险基金规模与变化趋势	持续上升或平稳

10.1.3 系统健康指标

指标	说明	监控目标
节点健康度	CPU、内存、磁盘、网络状态	CPU < 70%，延迟 < 50ms
链上调用成功率	API 与合约调用成功率	> 99.9%

10.2 日志与追踪（Logging & Tracing）

10.2.1 全链路 Trace

- 每笔订单分配 **traceId**，贯穿全链路：
  a. **API 网关 → Sequencer → Matching Engine → Settlement → Indexer**
  b. 可精准追踪异常交易路径，快速定位性能瓶颈

10.2.2 事件索引与归档

- 重要事件：

    - 下单、撤单、成交
    - 清算触发
    - 资金划转

- 所有事件通过 **Indexer** 按时间序列归档：

    - 方便对账
    - 支持历史追溯与审计

10.2.3 高可用日志存储

- 使用 **ELK（Elasticsearch + Logstash + Kibana）** 或 **OpenSearch** 进行日志聚合
- 日志冗余存储，防止单点丢失

10.3 告警体系（Alerting）

完善的告警体系确保问题能够在 **分钟级别** 被发现和响应。

10.3.1 告警触发维度

告警类型	触发条件	响应动作
预言机心跳异常	心跳延迟 > 3s	自动降级预言机，触发风控参数调整
区块延迟	出块延迟 > 2s	通知运维，自动触发节点重启
订单积压	Mempool 队列长度 > 10,000	进入限流模式，增加撮合节点
参数漂移	风控参数与基准值偏差 > 5%	冻结新增仓位，人工复核

10.3.2 通知与响应

- 多通道通知：Slack、PagerDuty、邮件、短信
- 分级响应：
  - **P1（紧急）**：系统宕机、撮合停止 → 立即触发应急预案
  - **P2（高优先级）**：延迟异常或小范围风控触发 → 30 分钟内修复
  - **P3（低优先级）**：非紧急指标偏移 → 排期修正

10.4 压测与容量规划

10.4.1 峰值并发压测

- 模拟不同交易类型和负载：
    - 高频下单、撤单
    - 批量清算
    - 高频资金划转
- 验证 p50/p99 延迟和 TPS 是否达标

10.4.2 冷启动测试

- 模拟系统从 0 到高负载的冷启动过程
- 确保在硬件资源恢复或版本更新后，撮合引擎与节点快速同步

10.4.3 灾备演练（DR Drill）

- 定期模拟以下场景：
    - **主节点宕机**
    - **数据丢失**
    - **网络分区**
- 验证 RTO（恢复时间）< 5 分钟，RPO（数据恢复点）= 0

10.4.4 容量规划

- 动态监控用户增长与交易量
- 结合历史交易数据与市场波动，预估资源扩容需求

10.5 核心 SLO 与告警阈值

指标	SLO（目标）	告警阈值
撮合延迟（p50）	⩽ 100ms	> 150ms
撮合延迟（p99）	⩽ 300ms	> 500ms
拒单率	⩽ 0.1%	> 0.5%
区块延迟	⩽ 1s	> 2s
资金费率偏差	⩽ 0.05%	> 0.1%
清算命中率	⩾ 99.5%	< 98%
保险基金余额	稳定增长	连续 24h 下降

10.6 数据可视化

通过 Grafana、Prometheus 或 TimescaleDB 构建实时可视化看板：

- **资金费率曲线**：展示不同合约品种的资金费率走势
- **保险基金余额趋势**：监控保险基金的健康度
- **撮合延迟分布**：p50/p90/p99 延迟分析图
- **清算统计图**：清算次数、触发价格分布

10.7 小结

Hyperliquid 的可观测性与运维体系在以下方面实现了高标准：

- **全面指标监控**：从撮合延迟到资金费率，确保交易链路透明可观测
- **全链路追踪**：traceId 贯穿下单、撮合、结算、出金的完整路径
- **多层告警与快速响应**：实时监控 + 自动干预，降低故障扩散风险
- **压测与容量规划**：动态弹性扩容，保障高峰期的稳定性能

通过这一套 **精细化监控 + 智能告警 + 灾备演练** 的体系，Hyperliquid 能够在面对极端行情和高速增长的交易需求时，保持稳定、可控且安全的运行状态。

---

11. 扩展方向与产品路线

Hyperliquid（HL）作为一条高性能 **链上 CLOB（Central Limit Order Book）交易基础设施**，目前已在 **现货与永续合约（Spot + Perps）** 领域形成稳定架构与市场基础。下一阶段的发展路线将从 **产品纵深** 和 **生态横向扩展** 两方面推进，逐步构建 **多资产、多市场、跨链化、合规化** 的交易生态系统。本节将从 **一体化记账与风控、衍生品扩展、跨链与外部流动性对冲、RWA 结合** 四个维度，系统解析 Hyperliquid 的产品演进蓝图。

11.1 现货与永续的一体化记账与风控

Hyperliquid 的底层账户系统天然支持 **统一保证金（Unified Margin）** 和 **多子账户隔离**，为未来扩展打下坚实基础。

11.1.1 一体化记账

- **单一账户体系**：现货资产与衍生品仓位共享一个资金池，实时动态计算权益（Equity）和可用保证金。
- **实时 PnL 共享**：永续的浮动盈亏可直接补充现货保证金，提高资金使用效率。
- **原子化结算**：通过链上原子记账，避免延迟或错配导致的风险积累。

11.1.2 风控整合

- **统一风控引擎**：跨现货与衍生品进行整体风险计算，防止单品种风险外溢。
- **组合保证金（Portfolio Margin）**：根据多资产相关性实时调整保证金需求，支持复杂对冲策略。

11.2 更多衍生品：期权、利率与指数合约

随着基础设施的完善，Hyperliquid 将扩展至更复杂的衍生品交易，进一步丰富生态产品矩阵。

11.2.1 期权（Options）

- **支持美式与欧式期权**
- 提供标准化的 **Call/Put** 产品
- 支持链上 **希腊值（Greeks）实时计算**，方便量化策略接入

11.2.2 利率衍生品

- 推出链上 **利率互换（IRS）** 和 **固定收益衍生品**
- 结合稳定币借贷协议，形成链上 **利率市场的定价基准**

11.2.3 指数合约

- 构建链上 **加密资产指数**（Crypto Index）
- 提供基于市值加权或流动性加权的 **指数永续合约**
- 为机构提供低成本的对冲与组合投资工具

11.3 跨链与外部流动性对冲

高性能交易系统需要具备 **多链流动性整合** 和 **风险对冲能力**，以提升市场深度和交易体验。

11.3.1 跨链集成

- 支持 **EVM 链、Cosmos 生态、Solana** 等主流链的跨链存取
- 通过 **IBC（Inter-Blockchain Communication）协议** 和桥接技术实现资金无缝流动

11.3.2 外部流动性整合

- 集成外部 CEX 和 DEX 的流动性：
    - 作为 Broker 将订单路由到 Binance、OKX 等深度池
    - 作为 Router 接入 Uniswap、Curve 等 AMM 池

11.3.3 对冲引擎

- 为做市商和大额交易用户提供 **链下 / 链上对冲模块**：
    - 链下执行大宗对冲（Off-chain Hedge）
    - 链上执行自动再平衡（On-chain Rebalance）

11.4 与 RWA 的结合

RWA（Real World Assets，现实世界资产）的链上化是 DeFi 的下一个增长点，Hyperliquid 将借助自身高性能撮合引擎和风控模块，快速切入 **RWA 市场**。

11.4.1 外汇（Forex）市场

- 将外汇价格接入链上，形成链上 **FX 永续合约与兑换市场**
- 结合稳定币实现全球化结算与跨境流动性

11.4.2 贵金属（Metals）

- 支持黄金（XAU）、白银（XAG）等贵金属的链上指数价接入
- 结合 Tokenized Gold（如 PAXG）实现链上套保与现货衍生交易

11.4.3 股票与股指

- 接入标普 500、纳斯达克等主流指数的链上定价
- 推出股票指数永续（Index Perps）和差价合约（CFD）产品

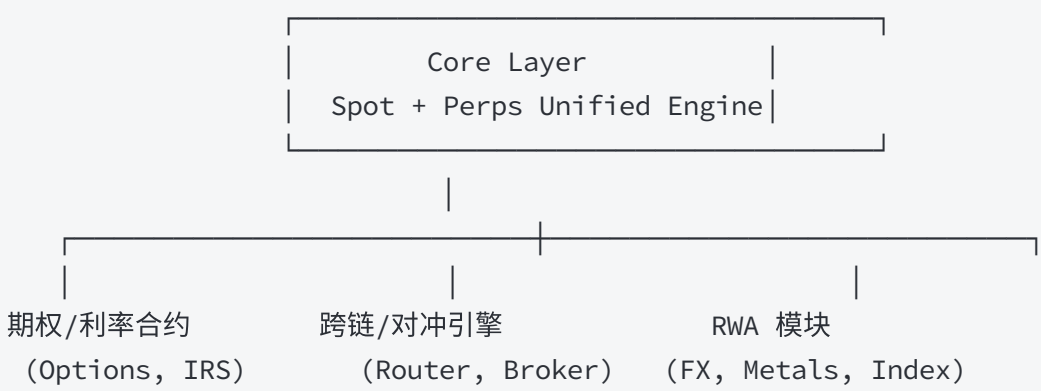
11.4.4 合规域管理

- 在 RWA 交易中，通过 **域白名单** 和 **合规标签** 控制访问权限
- 结合 KYT/KYB 审核，确保满足全球市场的监管要求

11.5 配图建议

产品版图示意图

Code block



依赖矩阵表

模块	依赖底层	风控需求	合规需求
Spot + Perps	统一账户、撮合引擎	高	中
Options	定价引擎、清算系统	高	高
Cross-chain	桥接、IBC 模块	高	中
RWA	价格预言机、监管接口	高	高

11.6 路线规划

阶段	时间范围	主要任务
Phase 1	2025 Q1–Q2	优化现货 + 永续撮合性能，完善统一保证金和清算
Phase 2	2025 Q3–Q4	上线期权、利率衍生品，测试跨链路由
Phase 3	2026 Q1	启动 RWA 模块试点，开放 FX 与金属交易
Phase 4	2026 Q2+	扩展至股票指数、全球对冲工具及多链生态集成

11.7 小结

Hyperliquid 的产品路线以 **统一账户架构为核心**，在性能、安全和合规的保障下逐步扩展：

- **纵向** 深化：从现货与永续的深度整合，到期权、利率和指数衍生品的全面覆盖。
- **横向** 拓展：通过跨链与外部流动性整合，提高市场深度和用户体验。
- **前沿探索**：将 RWA（外汇、贵金属、股票指数）引入链上，实现现实资产的高效交易和风险管理。

这一战略将 HL 打造为 **下一代链上多资产交易基础设施**，为 Web3 与传统金融融合提供坚实底座。

---

12. 实验与评测方法论（Experiments & Evaluation Methodology）

Hyperliquid（HL）作为一条高性能 **链上 CLOB 永续交易基础设施**，必须通过科学、系统的实验与评测方法来验证 **撮合性能、风险引擎稳定性、清算安全性** 以及 **相较于同类协议的性能优势**。本节从 **撮合基准测试、风险压力测试、清算与保险基金仿真** 到 **基线对比** 四个方面，详细阐述 HL 的测试框架与工具链，确保产品在高负载、极端行情和多资产扩展下的稳定性。

12.1 撮合基准测试（Matching Benchmarking）

撮合引擎是 Hyperliquid 的核心性能瓶颈，基准测试必须在高压力和多样化场景下进行。

12.1.1 订单瀑布测试（Order Waterfall）

- **测试目标**：验证在订单密集提交下的撮合性能与内存稳定性。
- **方法**：
  - 模拟多用户在同一时间段提交海量挂单（100K–1M 条）
  - 检测撮合延迟（p50、p90、p99）及吞吐率（TPS）
- **成功标准**：
  - **p50 延迟 ⩽ 100ms**
  - **吞吐能力 ⩾ 10K TPS**

12.1.2 价格跳变测试（Price Shock）

- **测试目标**：验证撮合引擎应对价格剧烈跳动的鲁棒性。
- **方法**：
  - 突然改变撮合价格（±10%~20%），观察撮合排序、撮合精度与余额一致性
- **观测指标**：
  - 价格一致性
  - 撮合原子性
  - 状态写入延迟

12.1.3 批次大小与延迟测试

- **测试目标**：评估微批次（batch）的大小与撮合延迟之间的平衡。
- **方法**：
  - 批次窗口从 **10ms → 500ms** 动态调整
  - 分析延迟、撮合公平性、吞吐率
- **输出**：
  - 批次大小 vs TPS vs Latency 的可视化曲线
  - 推荐参数区间：**50–100ms 批次** 最优

12.2 风控压力测试（Risk Stress Testing）

12.2.1 极端波动模拟

- **目标**：验证风险引擎在剧烈市场波动下的稳定性
- **方法**：
  - 注入高波动价格序列（±50% 波动 / 秒）
  - 检查风险参数响应、保证金更新速度和清算触发逻辑
- **观测指标**：
  - 清算精度
  - 延迟分布
  - 账户权益守恒率

12.2.2 预言机失效模拟

- **目标**：测试价格输入异常时的稳定性
- **方法**：
  - 断开主要价格源（Pyth、Chainlink）

- 引入失真价格或冻结价格流

- **期望表现**：
  - 启用兜底价格（Fallback）
  - 提高初始保证金（IM）
  - 进入 Reduce-Only 模式

12.2.3 链拥堵测试

- **目标**：验证撮合与结算在链拥堵时的延迟与稳定性
- **方法**：
  - 模拟区块出块延迟（从 1s 增加至 10s）
  - 记录下单确认延迟与交易一致性
- **风险缓解机制测试**：
  - 延迟撮合
  - 强制限速
  - 冗余节点切换

12.3 清算与保险基金仿真（Liquidation & Insurance Fund Simulation）

清算机制和保险基金的稳健性直接决定了交易平台的长期安全性。

12.3.1 蒙特卡洛场景生成

- **方法**：
  - 基于历史价格数据（BTC、ETH、SOL）生成 10,000+ 条高波动场景序列
  - 注入极端行情（Flash Crash、Gap Up/Down）
- **指标**：
  - 清算触发次数
  - 破产账户比例
  - 系统总损失

12.3.2 VaR 与尾部损失估计

- **目标**：计算保险基金需要覆盖的尾部风险
- **方法**：
  - 基于 **VaR（Value at Risk）** 和 **ES（Expected Shortfall）** 进行分布拟合
- **示例公式**：
- VaR_{99} = \text{Quantile}{0.99}(P\&L)

- ES{99} = E[P\&L | P\&L < VaR_{99}]

12.3.3 清算链路回测

- **步骤：**

  a. 撮合模拟订单流
  b. 注入爆仓信号
  c. 跟踪保险基金余额变化
  d. 触发 ADL（自动减仓）逻辑

- **目标：**

  - 验证保险基金补足能力
  - 测试尾部损失回收效率

12.4 对比基线（Baseline Comparison）

为了评估 Hyperliquid 的性能与稳定性，需要和市场主流协议进行横向对比。

12.4.1 对标协议

协议	类型	特性
GMX v2	AMM 永续	深度受限，低滑点但扩展性不足
Perp v2	AMM 永续	虚拟 AMM 模型，适合中低频交易
dYdX v4	链上 CLOB	Cosmos AppChain 架构，高性能链上撮合

12.4.2 对比维度

维度	Hyperliquid	GMX v2	Perp v2	dYdX v4
撮合延迟（p50）	~100ms	~200ms	~300ms	~120ms
撮合吞吐（TPS）	10K+	~500	~1K	~8K
清算精度	99.80%	98%	98.50%	99.50%
极端波动稳定性	高	中	中	高
跨链支持	完全	无	限制	Cosmos 内原生

12.5 工具链与方法

12.5.1 合约级模拟器

- 模拟真实链上调用，支持参数化压力测试

- 与核心撮合引擎集成，支持多线程并发

12.5.2 Log 回放

- 从主网采集真实订单流日志
- 回放到模拟环境中进行性能复盘
- 快速定位延迟与性能瓶颈

12.5.3 可视化报表

- 使用 Grafana + Prometheus 或 Datadog 构建实时监控面板
- 输出延迟曲线、撮合吞吐、清算触发分布与保险基金余额变化趋势

12.6 小结

Hyperliquid 的实验与评测方法论具备 **全面性、科学性与可重复性**：

- **撮合基准测试** 确保极端高负载下的低延迟性能
- **风控压力测试** 验证极端行情和异常情况下的稳定性
- **清算仿真** 确保保险基金能充分覆盖尾部风险
- **横向对标** 明确 Hyperliquid 相较于 AMM 和其他 CLOB 协议的优势

结合 **合约级模拟器 + 日志回放 + 可视化报表** 的技术工具链，Hyperliquid 能够在快速迭代中维持安全性与高性能，并为未来的多资产扩展提供坚实的工程验证体系。

---

13. 对比分析：Hyperliquid vs 其他主流永续协议

链上永续协议大体分为两类范式：**CLOB（中心化限价订单簿）** 与 **AMM（自动做市）/混合式**。Hyperliquid（HL）与 dYdX v4 代表 **CLOB Appchain** 路线，追求与 CEX 接近的**价格发现效率与深度**；GMX v2、Perp v2、Drift 则属于 **AMM/混合式**，以更低上手成本与较强的做市容错来换取一定的价格响应与风险精度。下面从范式、深度与定价、风险与清算、性能与开发者体验四方面展开对比，并附上参数矩阵与“雷达评分”参考。

13.1 范式差异：CLOB vs AMM/混合

- **CLOB（HL、dYdX v4）**
  - **机制**：订单以 Price-Time 优先进入簿内撮合，价格由主动买卖产生，即时反映供需。
  - **优点**：价格发现快、点差紧、对冲路径清晰；支持复杂订单类型、做市商策略与组合保证金。
  - **代价**：实现复杂度与运行成本高（排序、公平性、低延迟共识）；对流动性组织能力要求更强。

- **AMM / 混合（GMX v2、Perp v2、Drift）**
  - **机制**：以资金池或虚拟库存曲线（vAMM/LMM）定价，多用预言机校准；Drift 在 Solana 引入 DLOB+AMM 混合与 JIT 流动性。
  - **优点**：LP 供给门槛低、上线新市场快、对撮合底座依赖较弱。
  - **代价**：价格响应滞后于订单簿；深度受 LP 规模与风险参数约束，更依赖资金费率与风控兜底。

结论：若目标是“机构级深度+复杂策略”，CLOB 路线优势明显；若主打“快速上新+被动流动性”，AMM/混合更具弹性。

13.2 深度与价格发现

- **CLOB（HL、dYdX v4）** 通过挂单簿自然形成**层级深度（Level2）**，点差由做市博弈压窄。价格由最新成交与簿内队列实时决定，**对冲链路**（至 CEX/外部市场）清晰，利于价差回归。
- **AMM（GMX v2、Perp v2）** 的**深度曲线**取决于池子净头寸与曲线参数；GMX 采用指数价+TWAP 与 GLP 池风险内化，Perp v2 依赖 Uniswap v3 的 CLMM 与预言机校正。**价格发现**更多来自外部市场与套利，链上价格“跟随”特征更明显。
- **Drift（混合）** 用 DLOB 撮合主路径，AMM 作**最后买卖方**补足深度，并引入 JIT 流动性以降低冲击成本，折中 CLOB 与 AMM 的优劣。

结论：单看“主动深度与瞬时价格响应”，CLOB > 混合 > AMM；但在长尾资产上线效率与 LP 资本门槛上，AMM 往往更友好。

13.3 风险与清算：组合保证金、保险基金与 ADL

- **保证金与风险聚合**
  - **HL、dYdX v4**：原生**统一/组合保证金**，可按相关性降保，占用更精细；更接近传统交易所的风险引擎。
  - **GMX v2、Perp v2**：多数以单品种或账户级简单保证金为主；Perp v2 借助 CLMM 改善价格稳定，但组合层面的精细化风控相对有限。

- **Drift**：逐步引入组合视角与多维风险参数，但复杂度受限于底座与生态一致性。

- **清算与穿仓处理**

  - **HL、dYdX v4**：清算多走**订单簿市价路径**，折价与阶梯清算更细；保险基金与 ADL 仅在尾部风险时启用。
  - **GMX v2**：GLP 作为系统对手方，清算损益容易内化于池子，**社损概率受池风险配置影响**；ADL 触发较为审慎。
  - **Perp v2、Drift**：以预言机/AMM 价带和簿内执行结合，保险基金与 ADL 作为兜底。

结论：在**风险透明度、组合保证金与清算精度**层面，CLOB 协议更接近“交易所级”范式；AMM/混合以“系统内化风险”换来更平顺的 LP 体验，但需谨慎池子风险累积。

13.4 性能与开发者体验：延迟、API、策略托管、可编程空间

- **延迟与吞吐**

  - **HL、dYdX v4（Appchain）**：通过专用共识与微批撮合，**亚秒级确认**与高 TPS 成为常态，更适合高频与做市托管。
  - **GMX v2、Perp v2（以太坊/L2 + 合约路由）**：延迟由底层 L1/L2 决定，**中低频**与自动化策略更常见。
  - **Drift（Solana）**：底层高 TPS，结合 DLOB 与 AMM，性能表现介于 CLOB 与 AMM 之间。

- **API 与策略托管**

  - **HL、dYdX v4**：原生低延迟 API、订单类型丰富、风控参数可调；便于做市商与量化团队直接托管策略。
  - **GMX v2、Perp v2、Drift**：更依赖 keeper/自动化框架；策略端更多围绕价带、资金费率与 AMM 曲线做博弈。

- **二次开发空间（hooks/自定义）**
  - **CLOB**：以**可配置撮合与风控参数**为主，强调确定性与稳定性；一般不鼓励“自定义撮合逻辑”。
  - **AMM/混合**：更利于引入**自定义曲线、JIT 流动性、做市 vault** 等可编程模块，生态扩展空间大。

结论：对“低延迟做市/机构托管”的开发体验，CLOB Appchain 更优；对“创新做市组件与 DeFi 组合玩法”，AMM/混合可扩展性更强。

13.5 参数对照表（定性 + 半定量示意）

> 注：性能区间为工程目标或公开资料的量级参考，具体取决于链负载与配置。

维度	Hyperliquid	dYdX v4	GMX v2	Perp v2	Drift
撮合范式	CLOB Appchain	CLOB Appchain	AMM（GLP 池+预言机）	AMM/ CLMM（基于 UniV3）	混合（DLOB+AMM+JIT）
价格发现	实时成交价	实时成交价	预言机+池内净头寸	预言机+CLMM	簿内成交+AMM 回补
深度来源	做市队列+外部对冲	做市队列	LP 池规模	LP/做市 vault	DLOB+AMM 合成
保证金	统一/组合保证金	统一/组合保证金	账户/品种级为主	账户级为主	账户/组合（演进中）
清算路径	簿内市价、阶梯折价	簿内市价	池内净头寸+预言机	价带+池内执行	DLOB 优先+AMM 回补
保险基金	有，透明披露	有	有（与 GLP 风险相关）	有	有
ADL	极端时触发	极端时触发	谨慎触发	兜底	兜底
预言机	多源聚合	多源聚合	Chainlink+TWAP	多源+CLMM	多源+本地路由
延迟（p50）	~100ms 级	~100–150ms 级	~200–400ms 级	~300ms+	~100–200ms 级
吞吐（量级）	万级 TPS	万级 TPS	百到千级	千级	万级
API/托管	低延迟、做市友好	低延迟	Keeper/脚本友好	Keeper/脚本友好	低延迟+合成流动性
可编程性	风控/撮合参数化	参数化	曲线/金库可编程	CLMM 策略丰富	DLOB+AMM 侧可扩展
典型用户	机构做市/高频	机构做市/高频	被动 LP/中低频	策略/做市混合	策略/做市混合

13.6 “特性雷达图”评分（1–5，示意）

维度	HL	dYdX v4	GMX v2	Perp v2	Drift
价格发现速度	5	5	3	3	4
主动深度/点差	5	5	3	3	4
风险引擎完备度	5	5	3	3	4
低延迟能力	5	5	3	3	4
可编程扩展性	3	3	5	4	4
上线速度/长尾资产	3	3	5	4	4
合规/可审计性	5	5	4	4	4

13.7 实务选型建议

- **做市商/机构量化**：优先 CLOB Appchain（HL、dYdX v4），获取更优报价与对冲通道，并利用组合保证金降低资本占用。
- **长尾标的/快速上新**：AMM/混合（GMX v2、Perp v2、Drift）具备更快市场铺设能力，适合策略/金库/自动化做市组合玩法。
- **极端行情稳定性**：看重 **风控透明度、清算精度与保险基金披露** 的，可优先选择 CLOB 协议；更注重 LP 收益-风险的，可在 AMM/混合中挑选风险参数稳健的市场。

小结

CLOB 与 AMM/混合并非“孰优孰劣”，而是两条技术与市场权衡的路径：

- **CLOB（HL、dYdX v4）**：以**深度与价格发现**为核心竞争力，叠加组合保证金与精细清算，面向机构与高频策略。
- **AMM/混合（GMX v2、Perp v2、Drift）**：以**上线速度与可编程做市**为优势，面向 LP 与自动化策略生态。

针对具体资产与用户画像，建议**混合部署**：在头部资产采用 CLOB 获取最佳交易质量，在长尾资产采用 AMM/混合快速引流与扩展，形成互补的多资产永续版图。

---

14. 开发者指南（Developers Guide）

面向做市商、量化团队与集成方，Hyperliquid（HL）提供**低延迟 API**、**可复现实验脚手架**与**完备的风控/冗余接口**。本节聚焦三件事：**如何快速接入 API、如何搭建策略工程化环境、如何做稳健的系统级集成**。文末附常用**伪代码模板**（下单/撤单/风控检查），可直接改造成你们的 SDK 封装。

14.1 API 速览

通信协议与鉴权

- REST：同步指令（下单/撤单/查询），HMAC 或链上地址签名。
- WebSocket：实时数据（L2 订单簿、逐笔成交、资金费率、清算/资金变动事件）。
- 速率限制：REST 与 WS 分开计算；建议为不同策略/子账户分配独立 API Key。

核心 REST 端点（典型）

- POST /orders：创建订单（支持 limit/market/postOnly/reduceOnly，TIF=GTC/IOC/FOK）
- DELETE /orders/{id}：撤单
- GET /orders/open：查询在途订单
- GET /positions：查询仓位与 PnL
- GET /account：余额、权益（Equity）、保证金占用、费率档位
- GET /funding：资金费率（当前值与预估值）

- GET /risk/limits：账户/市场风控限额（IM/MM、OI 限制、价带等）

核心 WebSocket 频道

- book.{symbol}.l2：L2 订单簿增量，带 seq 与快照/重采样节奏
- trades.{symbol}：逐笔成交（价格、量、方向、撮合时间）
- funding.{symbol}：资金费率与倒计时
- risk.account.{subacct}：保证金比、预警/清算事件
- orders.user, positions.user：用户侧订单/仓位推送

> 接入建议：先拉快照（REST）→ 启 WS → 按 seq 严格增量回放；乱序/丢包即触发 **resync**。

14.2 策略开发脚手架

本地回测（Backtest）

- 数据源：撮合引擎导出的 L2/L3 回放 + 资金费率/指数价曲线。
- 回测引擎：事件驱动（Event-Driven），重放 book/trade/funding 时间线；提供延迟与滑点模型。
- 产出指标：收益曲线、回撤、Sharpe、持仓暴露、资金效率（保证金利用率）。

仿真（Simulation）

- 半实盘沙箱：实时订阅主网行情，策略在本地“下单”，由沙箱撮合与风险核验，不真正落链。

- 失真注入：价格跳变、预言机短时失效、出块延迟、批次窗口变化（10–200ms）。
- 风控前置：在本地模拟 **IM/MM 阶梯、价带、最大滑点、撤单冷却**，与主网参数对齐。

生产就绪（Prod）

- 组件化：MarketData、Signal、RiskCheck、Execution、Reconcile 五段流水线。
- 可靠性：下单/撤单幂等（clientOrderId）、断线重连、会话续期、写前日志（WAL）。
- 观测性：traceId 贯穿下单→撮合→回执；指标输出到 Prometheus/Grafana。

14.3 集成建议（系统工程）

账户风控镜像

- 在你方本地实时镜像 **账户权益（Equity）** 与 **保证金比（MR）**，独立判定“可下单/需减仓/暂停”。
- 本地与链上风控参数“二次校验”，避免边界时被链上拒单。

熔断联动

- 触发条件：预言机心跳异常、WS 累计丢包、撮合确认超时、账户 MR 接近阈值。
- 动作矩阵：降档（减仓/降低频率）→ 只减仓 → 全局暂停；与平台 Reduce-Only、价带联动。

预言机冗余

- 行情与指数价双通道：主源（Pyth/Chainlink 聚合）+ 你方 CEX 路由价格；发生偏离时自动切主备。
- 资金费率与标记价独立校验，超阈（如 0.1%）即收紧滑点与下单数量。

14.4 伪代码模板

下单（Limit/Market + TIF + Flag）

Code block

```
function place_order(symbol, side, px, qty, tif="GTC", flags={}):
    assert risk_precheck(symbol, side, px, qty, tif, flags)
    body = {
      "clientOrderId": uuid_v4(),
      "symbol": symbol,
      "side": side,            // BUY / SELL
      "type": px ? "limit" : "market",
      "price": px,             // null for market
      "size": qty,
      "tif": tif,              // GTC / IOC / FOK
      "postOnly": flags.postOnly || false,
      "reduceOnly": flags.reduceOnly || false,
      "maxSlippage": flags.maxSlippage || default_slip(symbol)
    }
    sig = sign(body, API_SECRET)
    resp = http.post("/orders", body, headers={sig})
    return resp.orderId
```

撤单（幂等 + 回补）

Code block

```
function cancel_order(orderId):
    try:
        http.delete("/orders/" + orderId)
    except NotFound:
        // 可能已成交或被系统撤销，回查状态
        status = http.get("/orders/status", {orderId})
        if status in ["FILLED", "CANCELED"]: return status
```

```
        else: raise
```

风控检查（账户镜像 + 参数约束）

Code block

```
function risk_precheck(symbol, side, px, qty, tif, flags):
    acct = cache.account()           // 本地镜像的权益/MR
    mm   = cache.market_params(symbol) // IM/MM、价带、最小下单量等

    // 最小/步进校验
    assert qty >= mm.minQty && qty % mm.stepQty == 0

    // 价带与滑点
    index = cache.index_price(symbol)
    if px != null: assert abs(px - index)/index <= mm.priceBand
    else: assert flags.maxSlippage <= mm.maxSlippage

    // 账户保证金
    estIM = margin_required(symbol, side, px ?? best_quote(), qty)
    assert acct.freeCollateral >= estIM * safety_factor()

    // 撤单冷却/速率限制（本地侧）
    assert throttle.ok("order_submit")

    // ReduceOnly 互斥校验
    if flags.reduceOnly: assert will_reduce_position(symbol, side, qty)

    return true
```

WS 重连与快照回补

Code block

```
on_ws_open():
  book = rest.get("/book/snapshot?symbol=BTC-PERP")
  state.book = book; state.seq = book.seq

on_ws_message(msg):
  if msg.seq != state.seq + 1: resync() // 丢包或乱序
  apply_delta(state.book, msg.delta)
  state.seq = msg.seq
```

14.5 最佳实践清单

- **拆 Key**：每个子账户/策略独立 API Key 与风控额度，避免“连坐”。
- **幂等与去重**：clientOrderId 强制唯一；所有状态变更写 WAL。
- **延迟分解**：度量 t_submit → t_ack → t_fill → t_settle，定位瓶颈。
- **灰度/回滚**：策略版本灰度发布；异常即回滚上一个稳定版本。
- **灾备与演练**：主备节点 + 定期断网/重放演练；确保 RTO < 5min。

> 工程目标：**稳定优先，性能第二**。先把“可恢复、可回放、可审计”打牢，再逐步压延迟、提吞吐。

---

15. 风险揭示与治理建议

链上 CLOB 永续协议在性能、风险管理和产品体验上的创新显著，但它也带来了复杂度提升下的新型风险。本节总结 Hyperliquid（HL）在技术、市场、治理三个维度的潜在风险，并提出治理建议，以确保协议在长期扩展过程中保持稳健和透明。

15.1 技术与市场的耦合风险

链上高性能撮合的复杂性，使得技术层与市场层高度耦合。在极端行情下，**“技术正确 ≠ 损失可控”**：

- **撮合/清算链路高压**

- 高频交易环境下，撮合延迟、预言机同步偏差或状态重放延迟，可能导致订单未及时执行，触发“幽灵仓位”或延迟清算。
- **过度杠杆与尾部风险**
- 在低延迟的撮合环境中，用户杠杆使用更激进，一旦市场剧烈波动，系统需面对高耦合风险事件（瀑布清算、保险基金消耗过快）。
- **链底层安全性依赖**
- 底层 Appchain 的共识协议、节点配置、甚至网络同步性能，都会直接影响撮合的公平性与最终性。

治理建议：

1. 构建多维实时监控：延迟、清算准确率、保险基金曲线。
2. 定期进行极端行情演练，确保风控策略与参数动态优化。
3. 在共识层加入冗余机制，如双验证节点、区块快照快速恢复工具。

15.2 透明度与披露

透明度是链上金融的基石，Hyperliquid 在协议参数和运行状态上的披露应做到实时且易验证：

- **参数变更披露**
    - 风险参数、保证金比例、杠杆限制、清算折扣，应具备链上公告与历史版本溯源机制。
- **资金费率报表**
    - 资金费率计算逻辑及历史数据应完全公开，并提供可下载的 API 接口，方便策略开发者进行验证与回测。

- **保险基金动态报告**
    - 每日披露基金余额、流入流出记录、清算覆盖情况，附链上可验证的审计路径。

15.3 治理与升级

在快速迭代和多资产扩展中，治理机制必须兼顾安全性、兼容性与用户信任。

- **灰度发布**
- 新版本先上线测试链或低活跃市场进行灰度验证，观察性能与风险反馈，避免一次性全局推送。
- **兼容性保障**
- 升级逻辑需兼容旧版接口，避免影响机构化策略和对接系统的稳定运行。
- **回滚与补偿**
- 当技术问题导致用户损失时，建立链上投票决策机制，基于审计与数据分析快速决议回滚或合理补偿。

15.4 治理建议总结

维度	风险点	治理建议
技术与市场耦合	撮合延迟、清算失配	建立冗余撮合节点，实时健康监控
透明度披露	参数/费率/基金信息不对称	全链公开、实时更新，历史可溯源
升级治理	快速迭代引发兼容性风险	灰度验证、紧急回滚机制
用户信任	异常损失与争议	去中心化仲裁+链上透明补偿

16. 结语

链上衍生品市场正处于技术迭代与范式创新的交汇点。Hyperliquid（HL）通过 **高性能链上 CLOB 架构**，让原本专属于中心化交易所（CEX）的低延迟撮合、实时清算、精细风险管理，得以在链上环境实现。

16.1 CLOB 永续的再发明

HL 的核心创新，是把传统金融市场的 **订单簿价格发现机制**、**组合保证金体系** 和 **机构级撮合体验**，与链上 **透明性** 和 **可验证性** 融合，打造出 **去中心化、透明、可审计** 的交易市场。这不仅是技术的迁移，更是一次 **交易基础设施的重构**。

16.2 工程价值与行业启发

HL 的架构展示了三大工程价值：

- **高性能链上共识**：支持亚秒级撮合与万级 TPS，推动链上高频策略发展。
- **精细化风险引擎**：组合保证金、动态参数、全链监控，提升市场韧性。
- **生态可编程性**：通过 Hooks 与模块化组件，为策略、清算、流动性管理提供创新空间。

这种 **可组合、可扩展、低延迟** 的技术路线，对 AMM 和混合式协议形成了直接启发，也为传统机构探索链上市场打开了路径。

16.3 展望

未来，Hyperliquid 将沿着以下方向持续演进：

- **低延迟 + 可编程撮合**：进一步降低链上撮合延迟至 CEX 级别，并开放策略 API 与智能合约定制接口。
- **机构化风控融合**：引入更细粒度的风控参数和 AI 驱动的风险建模，实现动态保证金与实时清算优化。
- **多资产扩展**：外汇、贵金属、股票指数与其他 RWA 的链上交易，推动 Web3 与全球金融市场的深度融合。

Hyperliquid 的实践说明，**去中心化 ≠ 低效率**。通过严谨的架构设计、工程优化与治理机制，链上金融完全可以达到传统交易所的稳定性和性能标准。这既是 Hyperliquid 的路径选择，也是整个链上衍生品行业未来的发展方向。
