Eigenlayer & EigenDA 技术详解

▼ Eigenlayer解决的问题

▼ 当前以太坊的问题

在以太坊上构建去中心化基础设施的开发人员面临着建立自己的经济安全的挑战。虽然以太坊为智能合约协议提供经济安全性，但桥**或排序器等基础设施需要自己的经济安全性**，以使分布式节点网络达成共识。

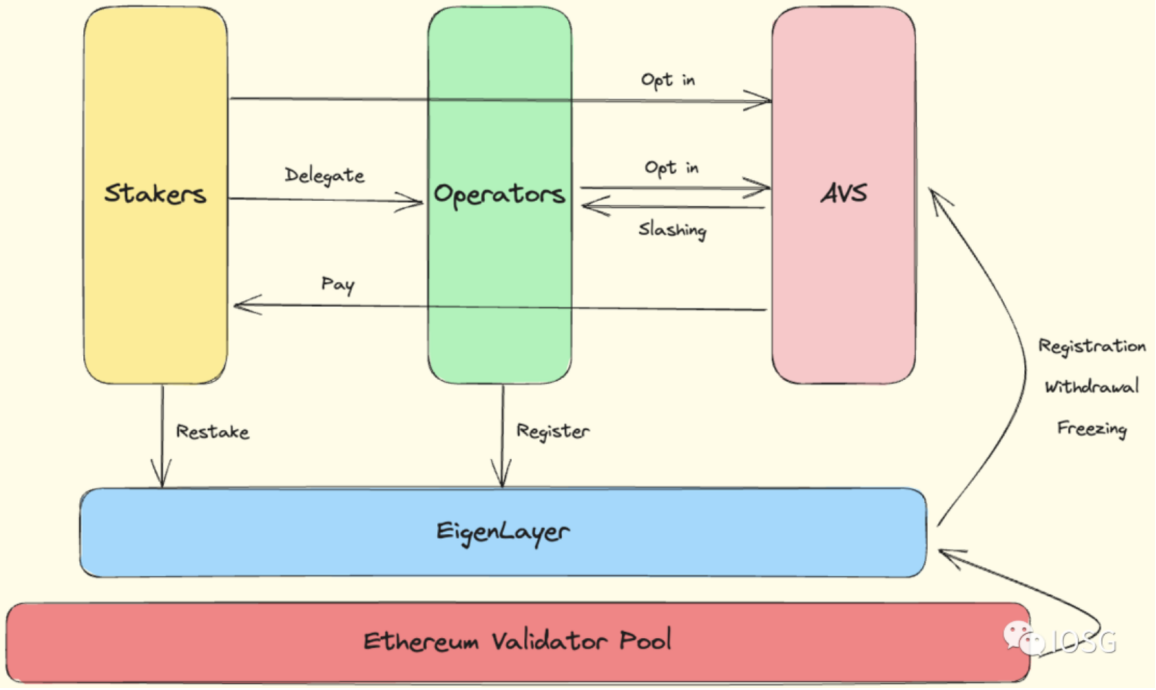
引入新的PoS网络很困难

1. 开发者找不到质押者
2. 质押者需要新的Pos网络大量原生代币，但是这些代币通常不稳定，收益也不稳定
3. 质押者要放弃其他奖励机会，譬如Ethereum提供的5%的奖励

▼ EigenLayer 如何解决

- 它充当连接质押者和基础设施开发人员的平台。
- 质押者可以使用任何代币提供经济安全。
- 质押者可以选择重新质押他们的股份，并为其他基础设施的安全做出贡献，同时获得原生 ETH 奖励。
- 通过重新获取，EigenLayer 可以汇集安全性，而不是对其进行碎片化。

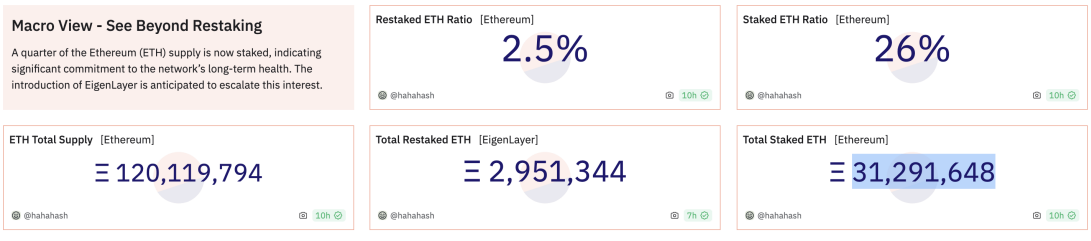
▼ 参与网络的角色



- **再质押者**。如果拥有以太坊质押做口，可以通过将提款凭证转移到 EigenLayer 来参与再质押，或简单地存入 stETH 等 LST 来参与。如果再质押者自己无法运行 AVS 节点，亦可将其做口委托给操作员。
- **Operators**。operators接受再质押者的委托，并运行 AVS 节点。他们可以自由选择为哪些 AVS 提供服务。一旦为 AVS 提供服务，就需要接受其定义的罚没规则。
- **AVS**。AVS 作为需求方/消费者，需要向再质押者付费，并获得其提供的经济安全。

▼ 数据指标

- https://dune.com/hahahash/eigenlayer

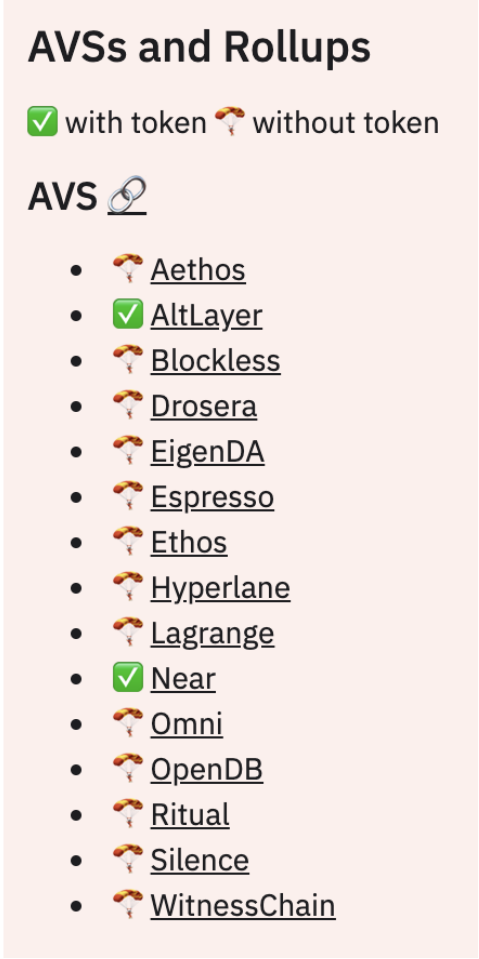


- 当前TVL: 110 亿 占总stake的TVL 2.5%

- 可以吸筹的市场空间1000亿，目前TVL还是项目方关闭了锁仓达到的成绩，还有空间

▼ Eigenlayer作为技术方案的优劣势

- 优势
  - 可以直接引入以太坊带来的安全性，避免自建网络需要考虑的经济模型稳定 **(最大优势)**
  - 有助于整个网络的冷启动，在Eigenlayer已经存在大量可以充当质押者和operators
    - 11w的质押地址，测试网6000+的operators
- 劣势
  - 目前AVS还处在测试网阶段，暂时还不成熟
    - https://goerli.eigenlayer.xyz/
  - 进入avs的项目不确定是否有准入门槛



▼ AVS(主动验证服务) 介绍

> 在 EigenLayer 生态系统中，以太坊验证者可以重新质押他们的信标链 ETH，以原生 ETH 或 LST 的形式接受委托，并运行特定于主动验证服务 （AVS） 的节点软件。数据可用性层（如 EigenDA）是 AVS 的一个典型示例。

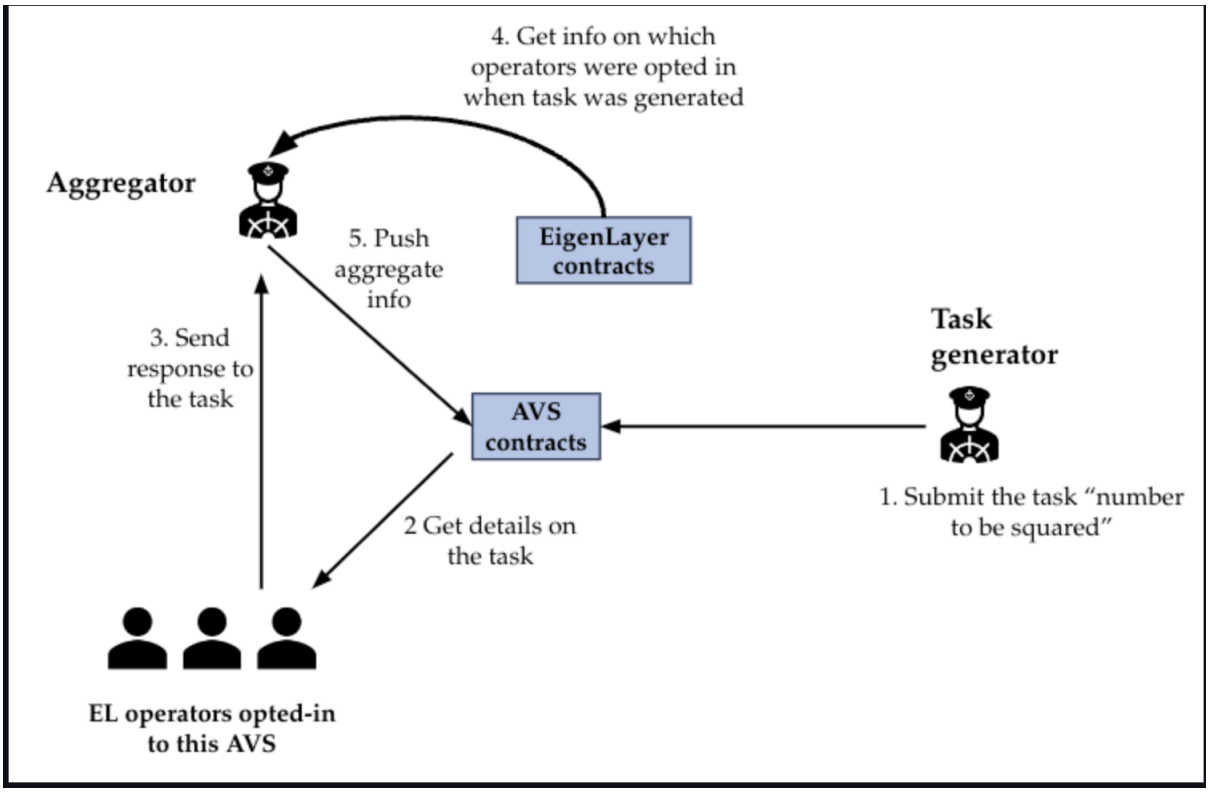
我的理解AVS是整个Eigenlayer作为一个质押的基础设施平台提供给开发者的开发套件。在大量质押者通过LST在Eigenlayer填充蓄水池，开发者可以基于AVS构建自己的链上应用，投放到市场让验证者运行你的链上应用，共享里面的蓄水池而达到共识层的经济模型稳定

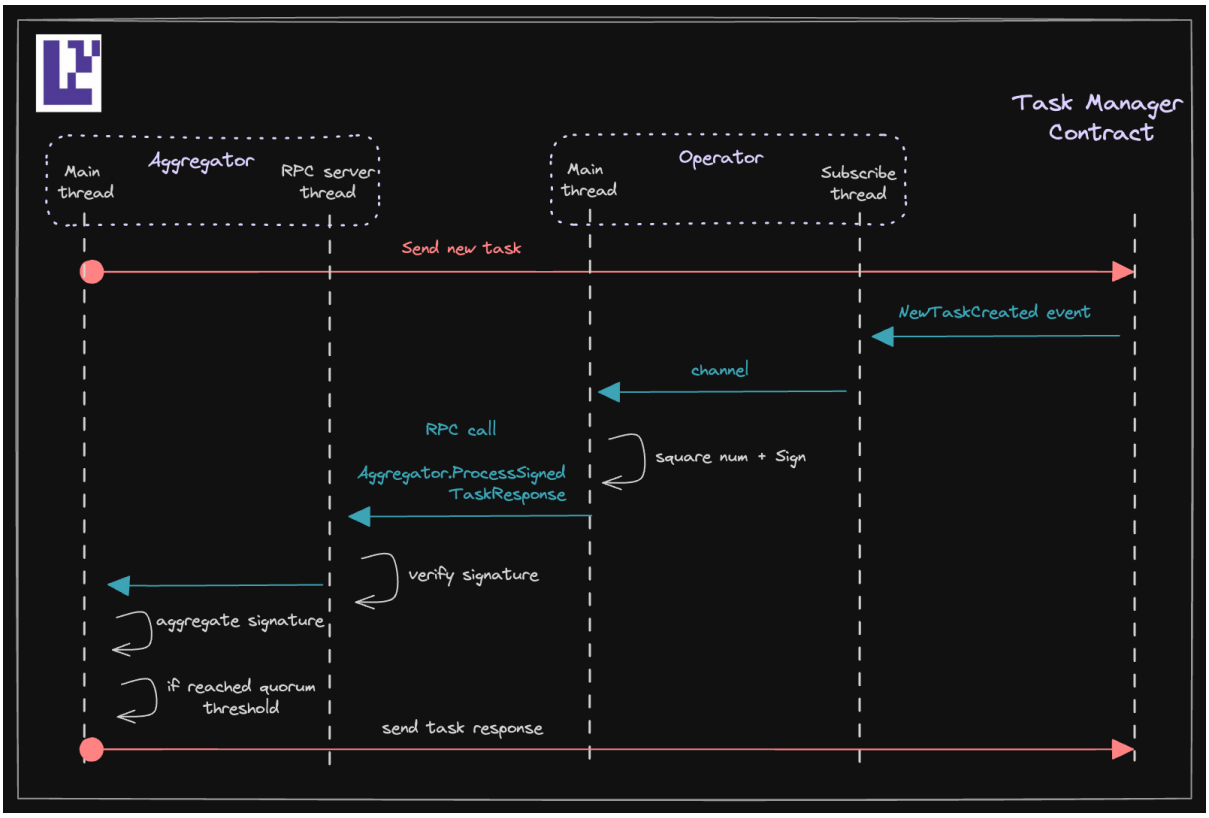
▼ 如何开发AVS

官方有个Demo可以参考 https://github.com/Layr-Labs/incredible-squaring-avs

主要分两部分需要实现

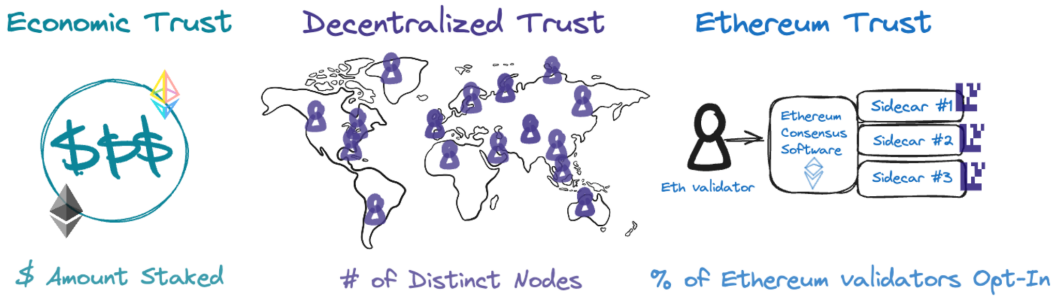
- 链上部分需要开发AVS的合约
  - 惩罚operators的削减逻辑
  - task的创建和响应逻辑
  - 发起挑战的逻辑，用于判断operators是否作恶
  - 管理operators的注册
- 链下部分需要开发让operators可以运行的程序
  - 基于EigenSDK开发https://github.com/Layr-Labs/eigensdk-go
  - 发起task任务：task就是需要达成共识的最小任务的原子
  - aggregator: 聚合所有op的响应，和EigenLayer的合约交互





设计AVS的四个问题

1. 定义AVS的“Task”
2. 继承什么样的信任



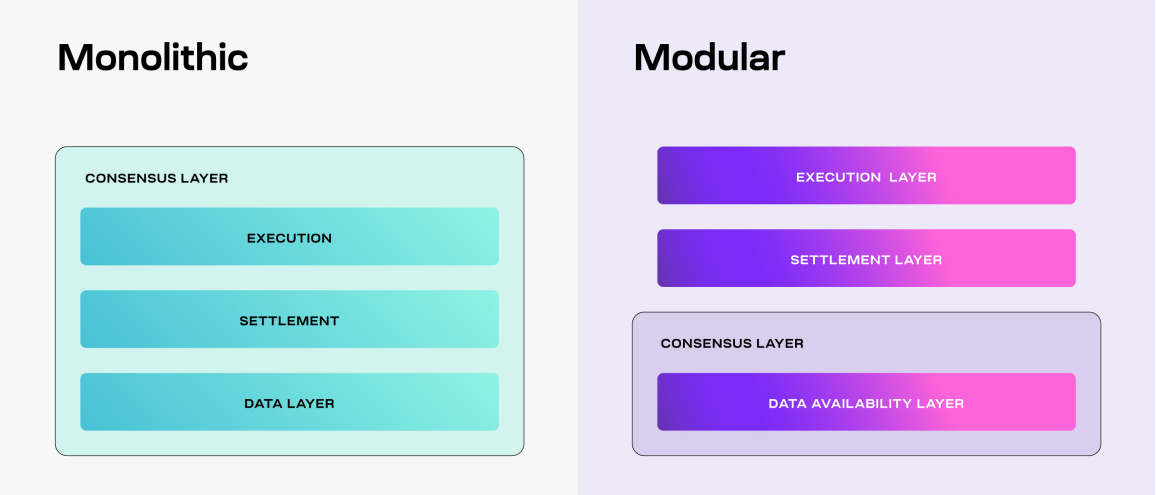
3. stake/slash 的条件
4. 工作量轻量级还是重量级

▼ AVS 项目案例 - EigenDA

▼ Why need Data availability

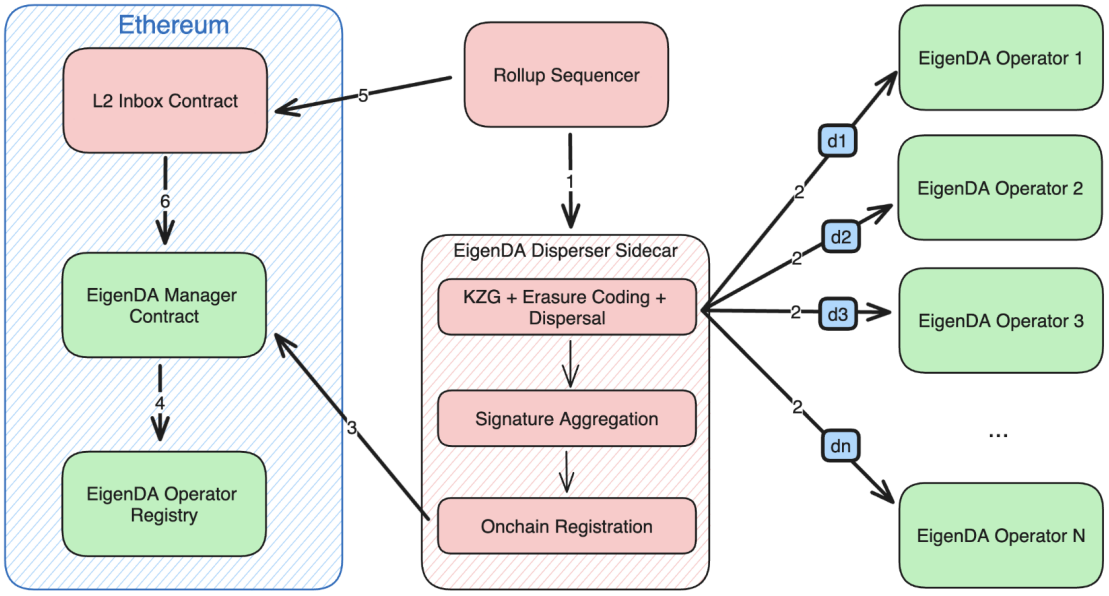
一般来说区块链都可以拆分成这三部分：执行层，结算层，数据可用层(DA 层)。

- 执行层: 执行层负责执行 transaction 并且根据有效的 transaction 正确地更新状态
- 结算层: 结算层提供给执行层一个环境去验证交易，解决欺诈问题。（在我理解是 p2p 网络的一个环境？）
- 共识层: 共识层负责最终决定 transaction 的执行顺序
- DA: DA 层保证 transaction data 可用，上面三层都需要依赖 DA。

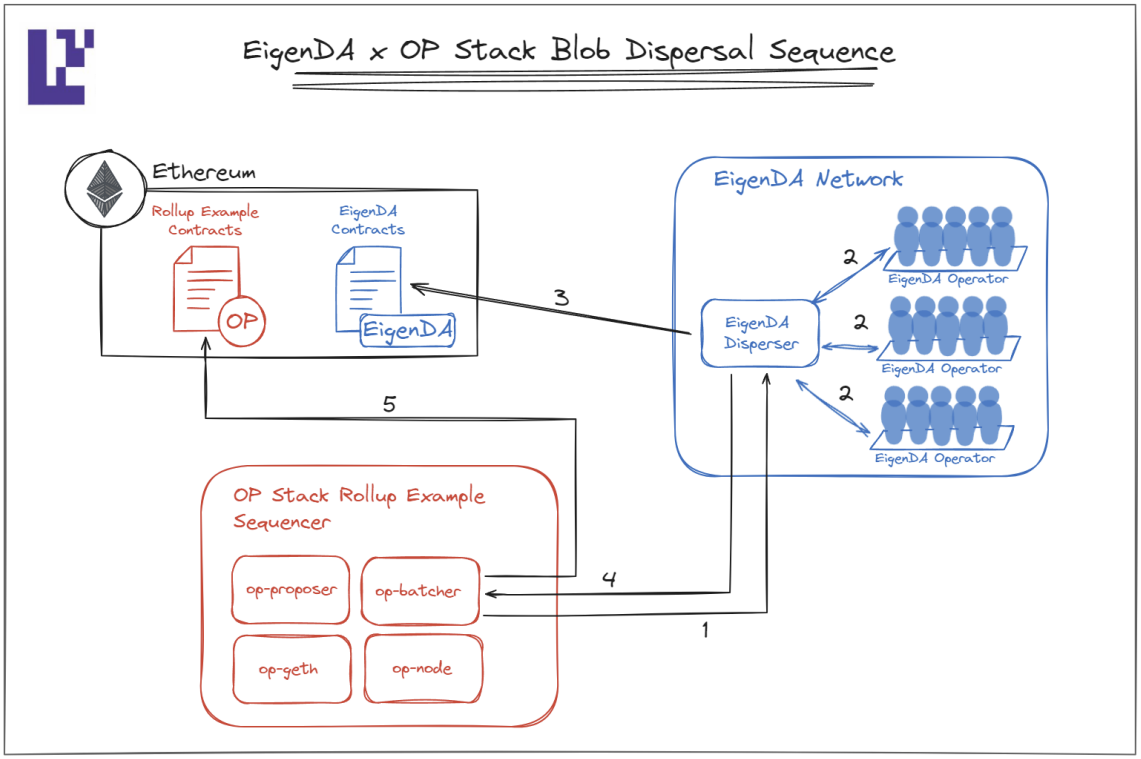


传统的区块链会在同一个共识层上面把所有的功能都实现，那么问题就是共识层需要执行很多不同的任务所以很难针对同一个功能点进行优化，所以不能变成一个高吞吐的系统。

▼ EigenDA工作原理



▼ Opstack集成



1. **Update DA writes**: modify how L2 transaction data is posted for DA (Op-batcher).

2. **Update DA reads**: modify how L2 transaction data is derived from DA (op-node).