OP-Stack 底层运行原理

登链社区 - Tiny熊

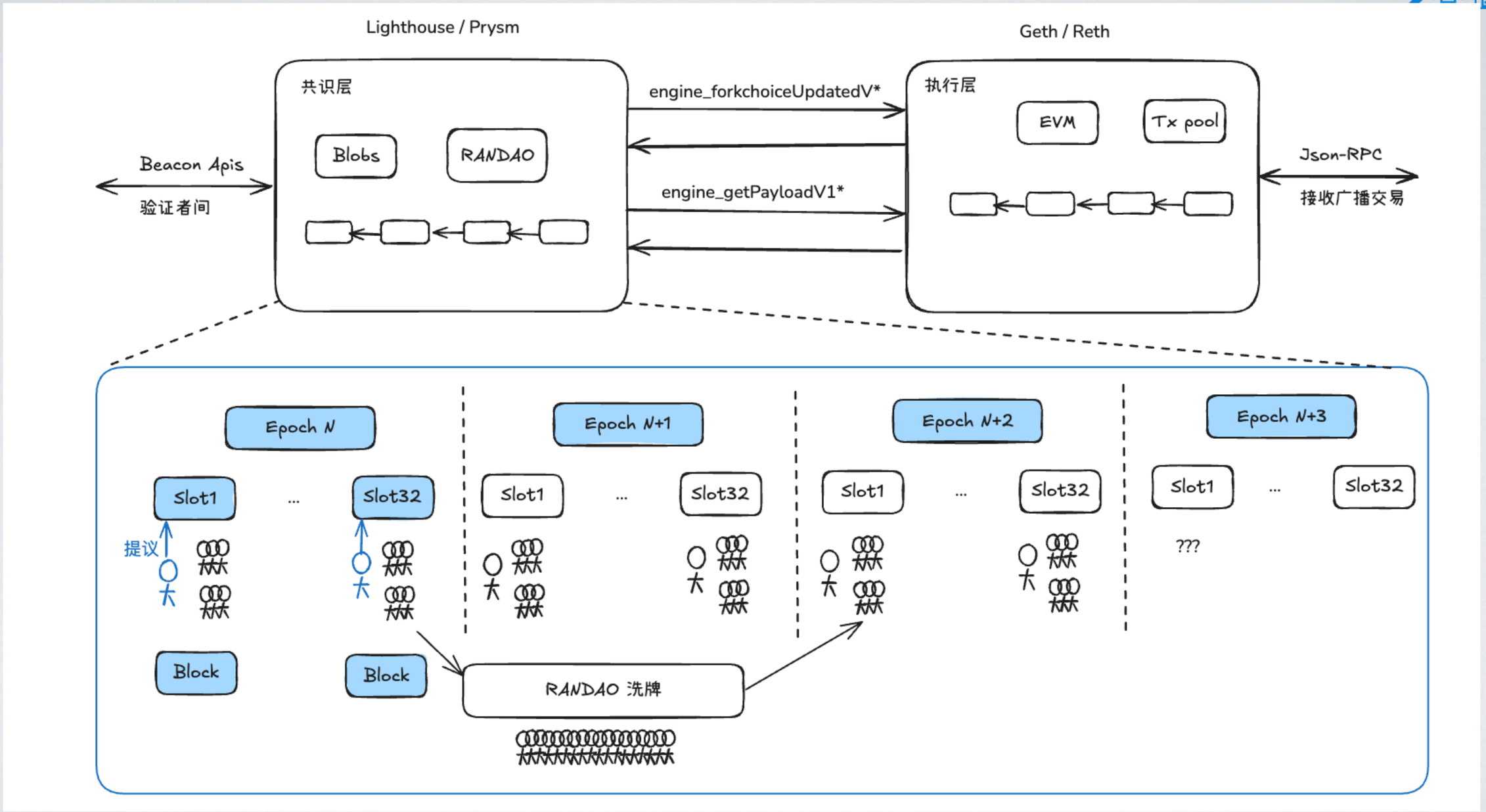
要点

- 以太坊的出块机制
- OpStack 主要模块
- OpStack 出块推导与跨链桥

以太坊 POS 共识

- POS 机制： 通过质押资金来保护网络安全，惩罚离线者和做恶者。
- POS 共识运行过程：
  - 以太坊质押 32 ETH 成为**验证者（pectra 升级：32 起步质押，最高质押 2048）**
  - 从验证者中**随机**选择一个作为**区块提议者**
  - 区块提议者从验证交易并创建区块
  - 其他的验证者（**见证者**）验证区块，并签名见证区块（确认该区块有效）
  - 当区块累计了足够多的见证投票后，成为**共识块**(达到最终确定性)

共识层通过 Engine API 驱动执行层

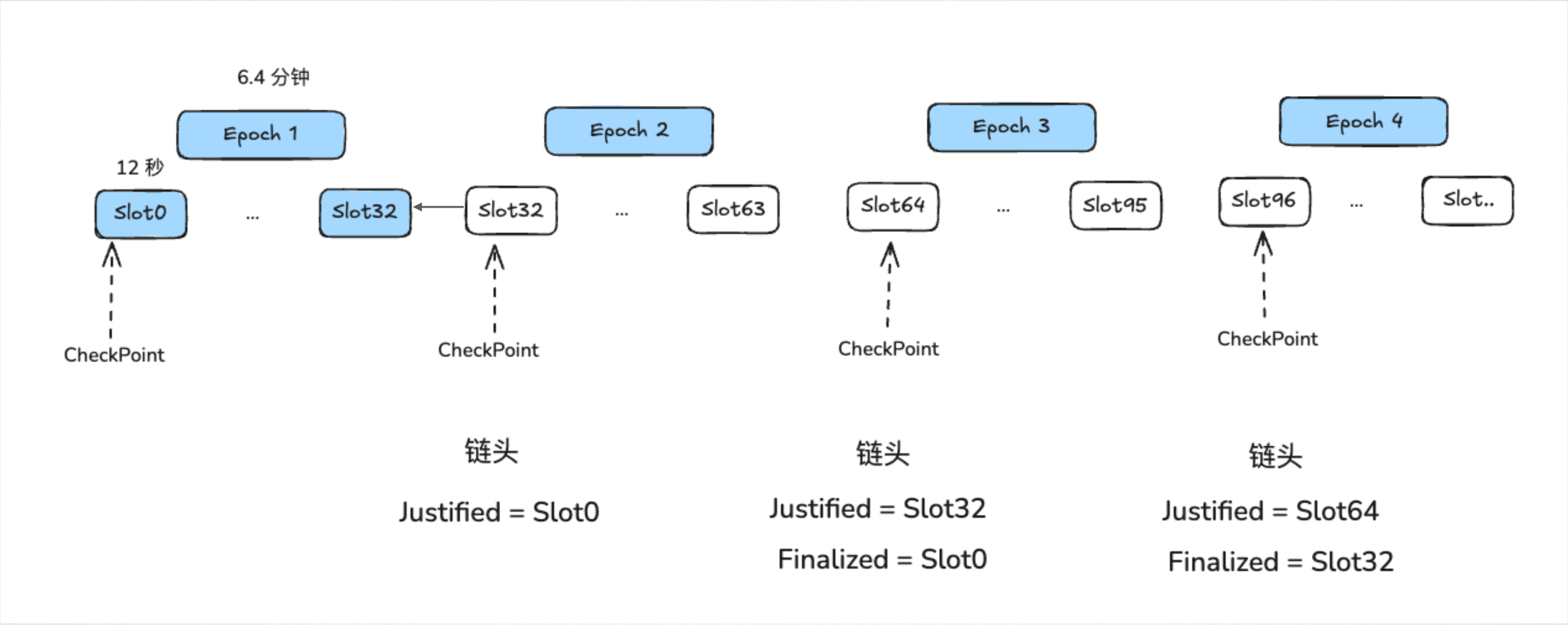


每个 slot 选出一个**提议者**和最多 64 个见证验证者 **委员会**，验证者的签名会聚合保存在信标链状态中

ETH 出块机制

- epoch: 质押周期，一个质押周期里面包含 32 slot
- slot: 一个 slot 里面可以放一个区块, 每 12s 出一个 slot，slot 是可以空
- 每个 epoch 开始时，会下一个 epoch 的提议区块者和验证者委员会，负责对区块提议和验证投票
- 每一个 slot 会有一个区块提议者来负责打包区块：
  - CL 将需基于的区块hash、timestamp、FeeRecipient、prevRandao 等信息，通过 engine_forkchoiceUpdatedV* 通知 EL 构建区块，然后通过 engine_getPayloadV* 获取区块执行相关信息（ExecutionPayload），记录到共识层区块
- 验证投票：
  - 验证者收到出块的信息后，调用 engine_newPayloadV2 让执行层验证区块，并调用 engine_forkchoiceUpdatedV2 更新链头，构建 Attestation 对象（源 checkpoint、目标 checkpoint、区块根）签名并广播。
  - 验证者委员会会随机选出少数几个作为聚合者，聚合签名放入下一个共识区块中。
  - 当产生分叉块，只能投其中一个分叉，不能同时投两个分叉块。

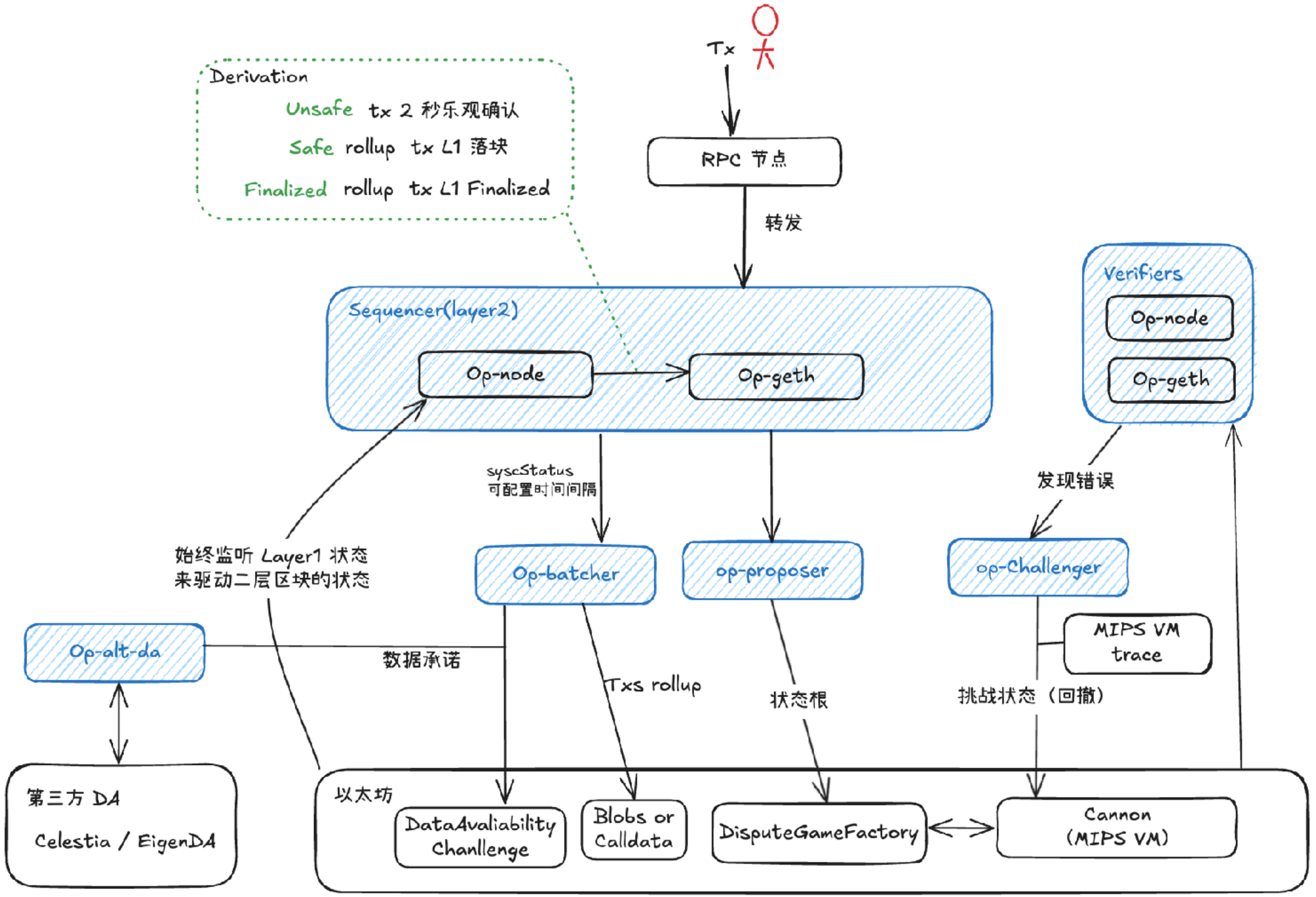
区块确定性



小于 finalized checkpoint 的slot 对应的区块，达到了最终确定性

OpStack

- OpStack 使用最多的 Layer2 开发框架
- 由 Optimism 团队开发，前身： OVM（Optimistic VM），OpStack 更接近原生 EVM、更好模块化和可定制


Derivation
Unsafe tx 2 秒乐观确认
Safe rollup tx L1 落块
Finalized rollup tx L1 Finalized
Tx
RPC 节点
转发
Sequencer(layer2)
Op-node
Op-geth
Verifiers
Op-node
Op-geth
syscStatus
可配置时间间隔
发现错误
始终监听 Layer1 状态
来驱动二层区块的状态
Op-batcher
op-proposer
op-Challenger
Op-alt-da
数据承诺
Txs rollup
状态根
MIPS VM
trace
挑战状态（回撤）
第三方 DA
Celestia / EigenDA
以太坊
DataAvaliability
Chanllenge
Blobs or
Calldata
DisputeGameFactory
Cannon
(MIPS VM)

OpStack
出块

关键组件与角色 I

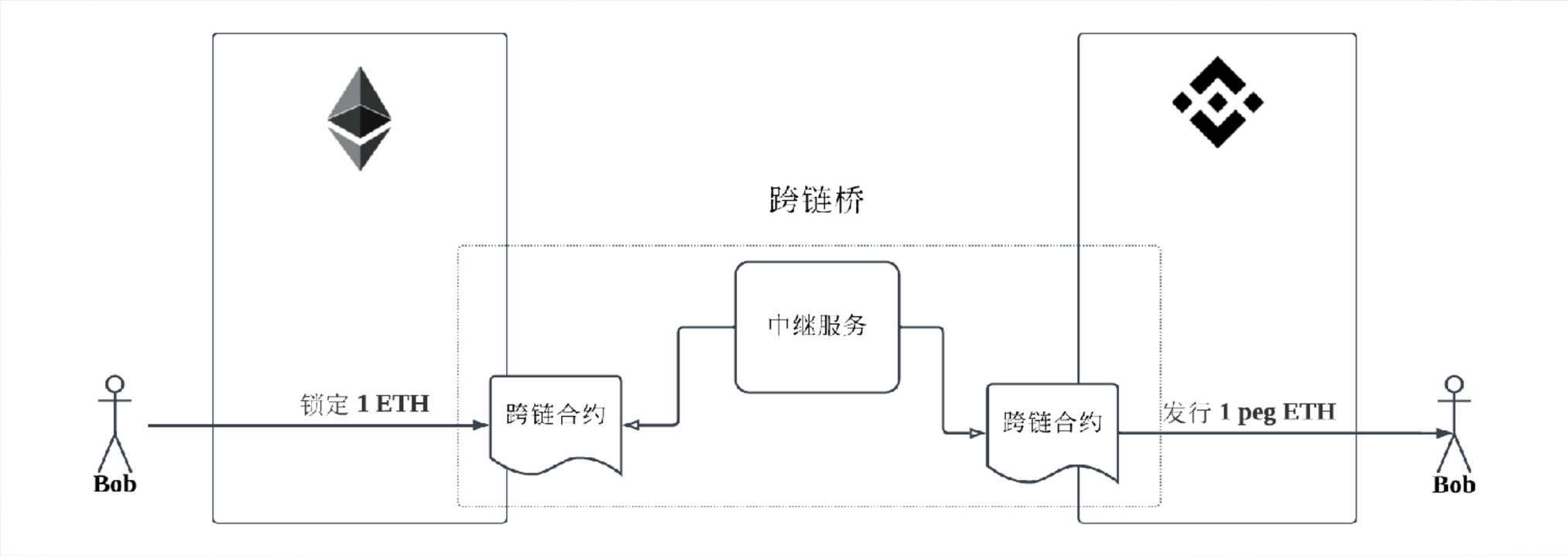
- OP Stack 智能合约：部署一套智能合约来管理自己的 Rollup 链
  - 代码库：optimism/packages/contracts-bedrock
- 排序器 Sequencer 节点 (分身-RPC 节点、验证者节点)： 收集用户交易，并处理
  - op-node: 类似共识层，不断监听 L1 区块数据，来指导 op-geth 出块，Layer2 的区块的状态（ unsafe、safe、finalized）是由 Layer1 来确定的
  - op-geth：修改自 geth 客户端、类似执行层，执行用户交易，出块

关键组件与角色 2

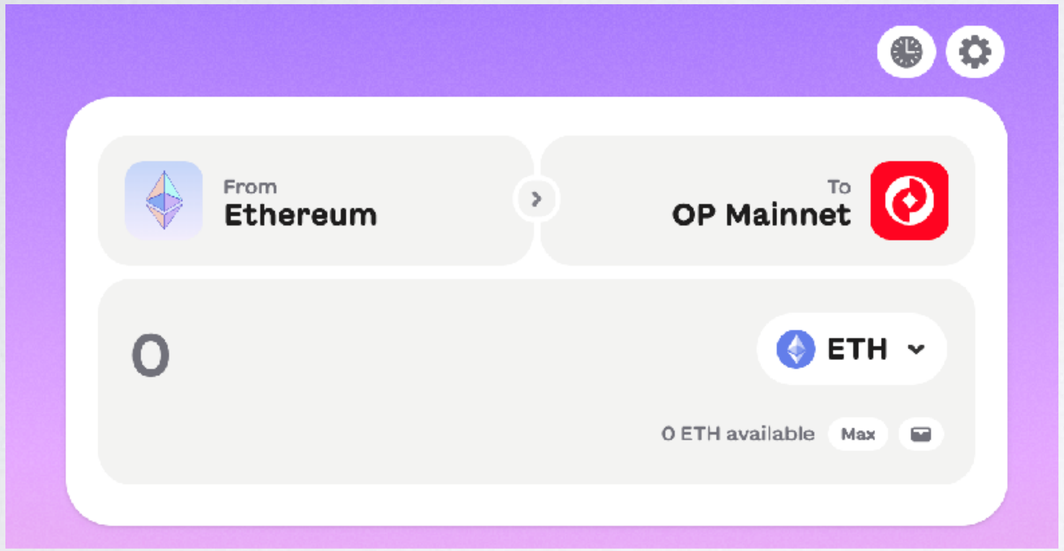
- Batcher（op-batcher）： 从 Sequencer 获取交易，将打包交易压缩发布到 Layer1 ：
  - 多数情况发布到 eip4844 blob
  - 如果 blob 手续费更高，也可以发布到 calldata
  - 如果提交到第三方 DA 的提交（op-alt-da ）， 则将数据承诺提交到 L1
- Proposer（op-proposer）：将交易结果（L2 状态根）发布到 L1 的提交状态根到 DisputeGameFactory ， 每次提交会创建一个挑战游戏，用户提款需要等待挑战。
- 挑战者（op-challenger）：挑战 Proposer 发布到 L1 的无效状态根，来维护网络安全
  - L2 状态执行过程翻译为 MIPS 指令，在链下 mipsvm 生成模拟执行需要的trace，交给 链上 Cannon 实现的 MIPS VM 合约中模拟 MIPS 指令运行，通过二分法找出错误的操作码。
  - 挑战成功之后，重新执行失败之后的所有交易。
  - op-challenger 惩罚机制还未实现

跨链桥实现

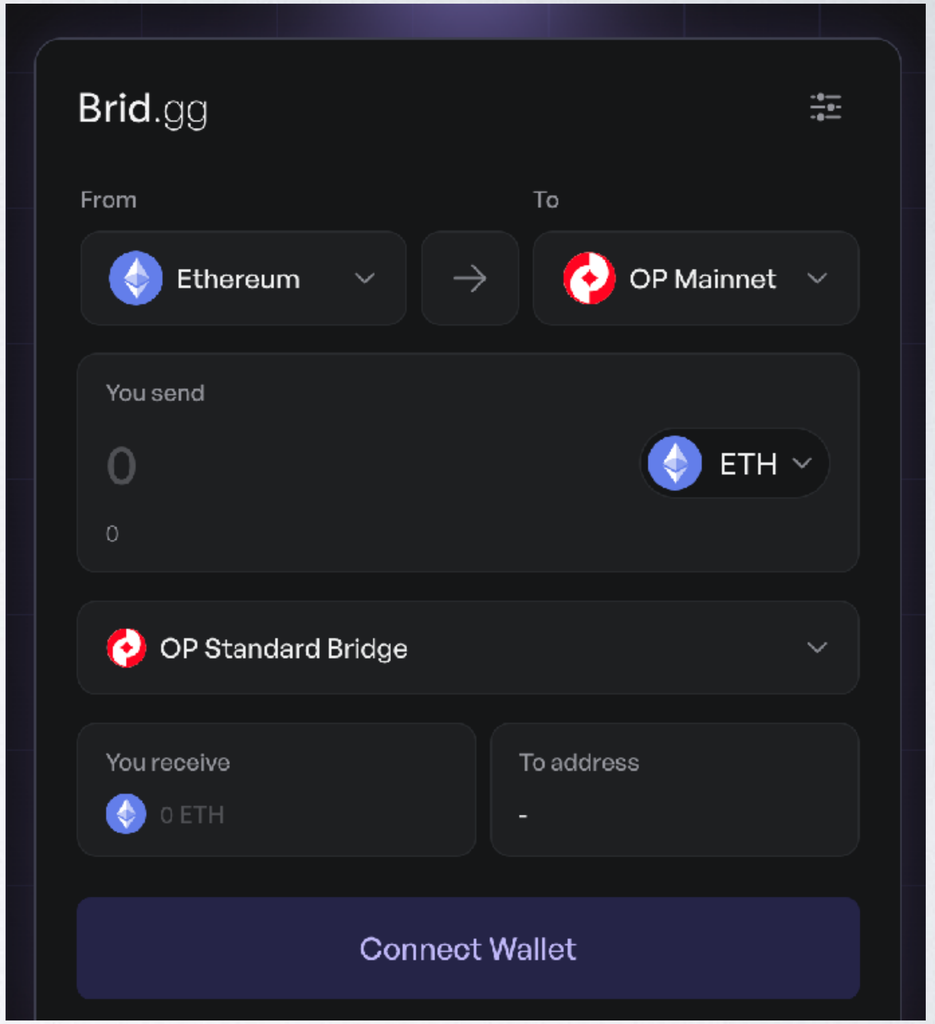
- 承载 L1->L2 充值和 L2->L1 提现的资产跨链
- 桥的通用工作方式：



跨链桥

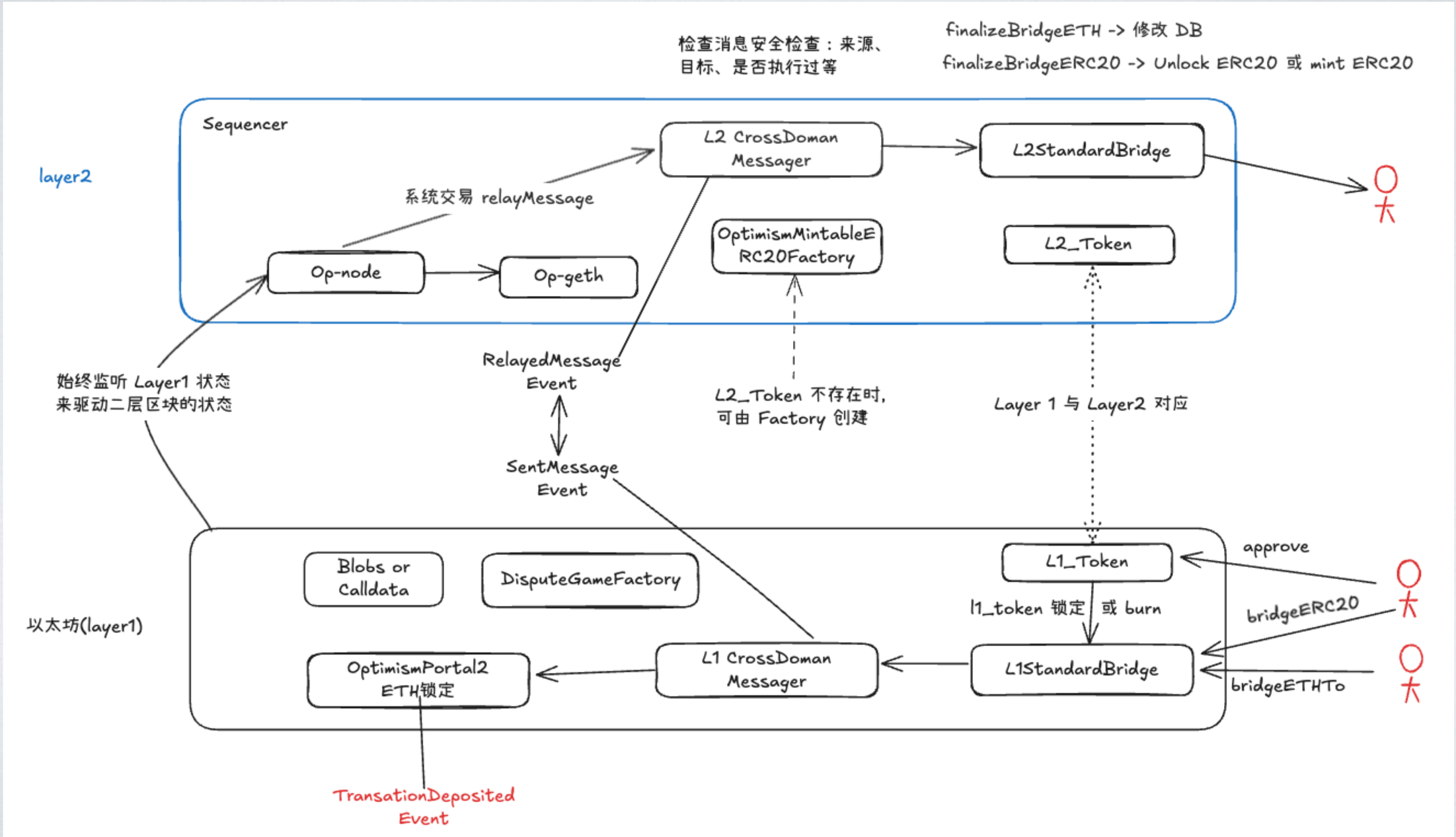


https://superbridge.app/



https://www.brid.gg/

桥 L1 -> L2 （充值）


检查消息安全检查：来源、
目标、是否执行过等
finalizeBridgeETH -> 修改 DB
finalizeBridgeERC20 -> Unlock ERC20 或 mint ERC20
Sequencer
layer2
L2 CrossDoman Messager
L2StandardBridge
系统交易 relayMessage
Op-node
Op-geth
OptimismMintableERC20Factory
L2_Token
RelayedMessage Event
始终监听 Layer1 状态
来驱动二层区块的状态
L2_Token 不存在时，
可由 Factory 创建
Layer 1 与 Layer2 对应
SentMessage Event
Blobs or Calldata
DisputeGameFactory
L1_Token
approve
以太坊(layer1)
l1_token 锁定 或 burn
bridgeERC20
OptimismPortal2 ETH锁定
L1 CrossDoman Messager
L1StandardBridge
bridgeETHTo
TransationDeposited Event

ERC20 充值流程

L1

- 准备：在 layer2 是否有对应的 l2_token， 若没有:
  - 可使用 OptimismMintableERC20Factory 创建
  - 或 自行创建一个 l2_token， 将 l2_token 转入到 L2StandardBridge
- 调用 L1 Token 给 L1StandardBridge 授权充值的金额
- 调用 L1StandardBridge 的 bridgeERC20()，锁定或 burn L1 token， 并触发相应的事件

Demo tx

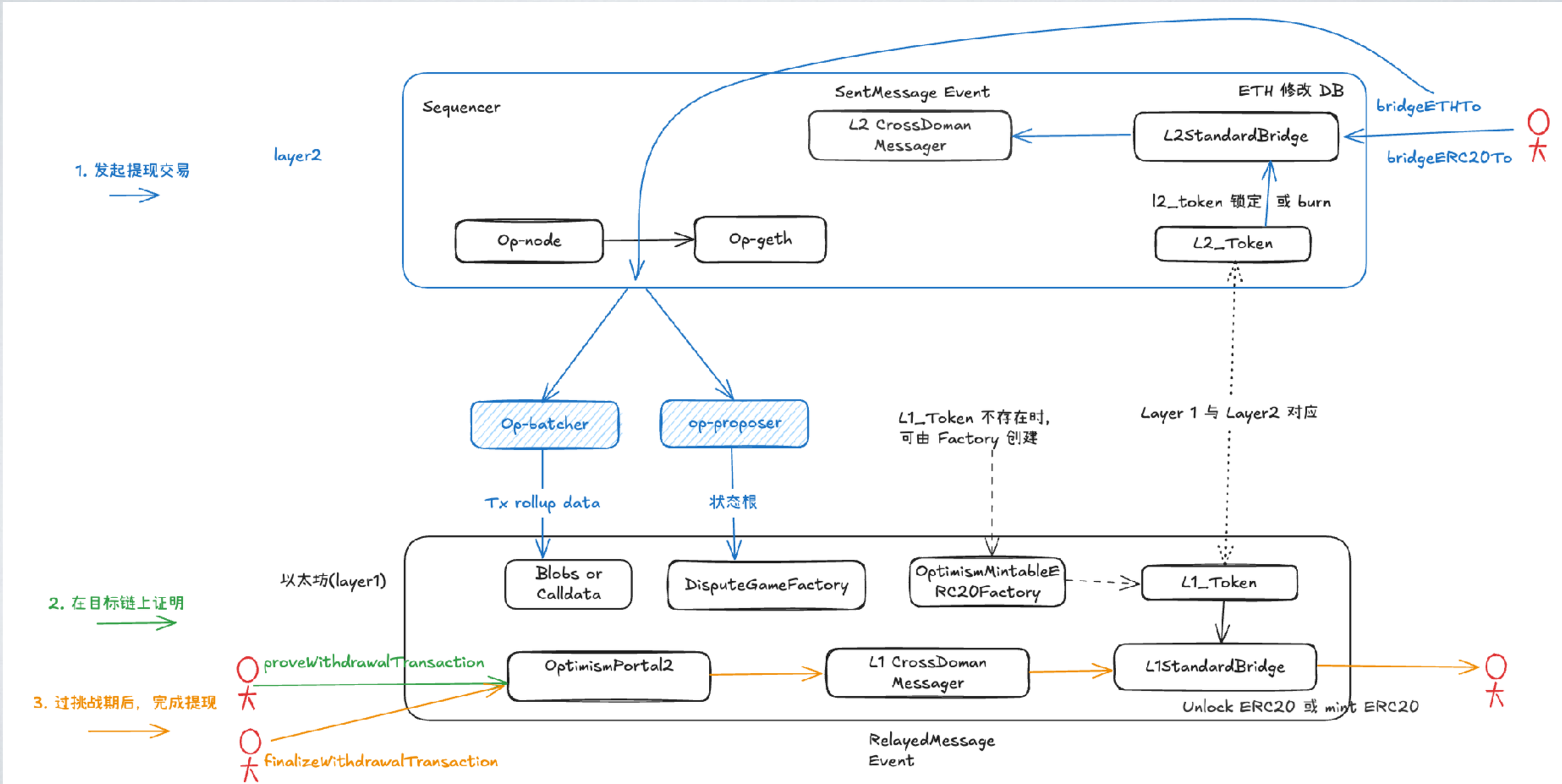
L2

- L2 的 op-node 会不断的获取 L1 的状态，当发现存款事件时
- 触发系统交易 L2CrossDomainMessenger 的 relayMessage， 对信息做一个安全性校验
- 调用 L2StandardBridge 的相应的 finalizeBriage 交易对用户转账
  - 如果是 OptimismMintableERC20，则 mint 给用户
  - 其他则解锁，转移给用户

Demo tx

脚本：https://github.com/lbc-team/hello_foundry/tree/main/sh

桥（L2 -> L1）提现


Sequencer
SentMessage Event
ETH 修改 DB
L2 CrossDoman Messager
L2StandardBridge
bridgeETHTo
bridgeERC20To
layer2
1. 发起提现交易
l2_token 锁定 或 burn
Op-node
Op-geth
L2_Token
Op-batcher
op-proposer
L1_Token 不存在时，可由 Factory 创建
Layer 1 与 Layer2 对应
Tx rollup data
状态根
以太坊(layer1)
Blobs or Calldata
DisputeGameFactory
OptimismMintableERC20Factory
L1_Token
2. 在目标链上证明
proveWithdrawalTransaction
OptimismPortal2
L1 CrossDoman Messager
L1StandardBridge
3. 过挑战期后，完成提现
Unlock ERC20 或 mint ERC20
finalizeWithdrawalTransaction
RelayedMessage Event

提现流程

L2

- 用户调用 L2StandardBridge 的 bridgeEth / bridgeERC20
  - 如果是 eth 直接修改用户余额 DB
  - 如果是 Token 执行锁定或 burn
- 交易通过 Op-batcher 将提现交易 rollup 到 L1
- 通过 op-proposer 将提现二层状态根（output root）提交到 L1 DisputeGameFactory

---

L1

- 等待提交状态根后，调用 OptimismPortal2 的 proveWithdrawalTransaction
  - 使用默克尔树 证明提款交易执行过，并且执行后的状态根已经提交到 DisputeGameFactory
  - 并等待挑战期
- 调用 OptimismPortal2 的 finalizeWithdrawalTransaction 完成提现

prove

demo

作业

- 实践一下:自己的 ERC20 跨链，查看调用 及 资金流转
- https://decert.me/quests/b81d49c5-277d-473b-9a34-eec6ce2d1681