摘要： 将数据存储在主链并进行共识，使得任何人可以在本地复现 Rollup 中的所有操作

译者注：

1.

原文中 Rollup
同时以单复数形式出现多次，在译文中统一确定为首字母大写的 Rollup。

Rollup 在当前的以太坊社区已经十分著名。在可预见的未来，Rollup 是以太坊实现扩容的关键解决方案 。但 Rollup 到底是什么？你能从这项技术身上得到什么，又该如何使用它？这篇文章将尝试回答一些关键问题。

背景知识：什么是一层扩容，什么是二层扩容？

实现区块链生态系统扩容有两种方式。第一种扩容的方式是让区块链本身拥有更高的事务处理容量。比如让区块变得 “更大” ，但 “大区块” 会让区块链的验证过程变得更为困难，而且可能会使节点更加中心化。为了避免这样的风险，开发者可以提高客户端软件的效率。更加具有可持续性的扩容方式是，使用分片等技术，让构建和验证区块链的工作分摊到许多不同的节点上—— “ eth2 ” 就在尝试使用这种方式升级以太坊。
第二种扩容的方式是改变使用区块链的方法。用户不需要直接把所有的活动都直接放在区块链主链上，而是在主链外的 “二层” 协议中执行大部分活动，并生成一个证明，证明链外发生的这一切活动都遵循规则。主链上部署一个智能合约，它只有两个任务：处理存取款，以及对上述证明进行验证。有多种方法实现证明和验证，但它们都有一个共同的特性，那就是在链上验证证明比在链下做原始计算的开销要小得多。

状态通道 vs plasma vs rollup

二层扩容主要有三种方案：状态通道，Plasma 和 rollup。三种方案代表了三种不同的范式，每种方案有自己的优缺点。所有的二层扩容大致都属于这三类（对于一些折衷的方案如何分类存在争议，例如 “ validium ” ）（编者注：中文译本）。

状态通道如何工作？

另请参阅 https://www.jeffcoleman.ca/state-channels ，statechannels.org

想象一下，Alice 向 Bob 提供了网络连接服务。作为交换，Bob 为上网产生的流量支付 0.001 美元/MB 的费用给 Alice。Bob 不需要在一层主链支付每笔费用，双方使用如下二层方案。

首先，Bob 将 1 美元（稳定币，或是等值的 ETH）存入一个智能合约中。为了向 Alice 支付第一笔款项，Bob签署了一张 "票据"（一条链外消息），“票据”上写着 “0.001 美元” ，并将其发送给Alice。为了支付第二笔款项，Bob 签署另一张写着 “$0.002” 的票据，并将其发送给 Alice。以此类推，每次付款都重复这个过程（译者注：状态通道参与方只需保留一个最新的状态变更，因此新的票据签署后，前一张票据作废）。当 Alice 和 Bob 完成交易后，Alice 可以将价值最高的票据包裹上自己的签名后，发布到主链上。智能合约会验证 Alice 和 Bob 的签名，验证通过后，将 Bob 的票据上标注的金额支付给 Alice，剩下的金额（译者注：1 美元减去前者）返还给 Bob。如果 Alice （出于恶意或技术故障）不愿意关闭通道，Bob可以发起一个提现挑战期（例如7天）——如果 Alice 在这段时间内无法提供 Bob 支付的票据，那么 Bob 就可以拿回之前存在智能合约里的所有钱。

状态通道技术很强大：广义的状态通道可以支持双向支付、实现智能合约（例如 Alice 和 Bob 在通道中签订金融合约），并具有可组合性（如果 Alice 和 Bob 之间有一个开放的通道，Bob 和 Charlie 之间也有一个开放的通道，那么 Alice 就可以和 Charlie 进行免信任的交互）。但是通道的作用是有限制的：不能向还没有加入通道的用户在链下发送资金；不能用于没有明确逻辑所有者的对象（比如 Uniswap智能合约）。此外，如果使用通道处理的事务比小额支付场景更复杂，需要锁定大量的资金在通道中。

Plasma 如何工作

另请参阅 Plasma原始白皮书，Plasma Cash要将资产从主链存入 Plasma 链，用户需要在主链将资产发送至管理 Plasma 链的智能合约。Plasma 链会给该资产分配一个新的唯一 ID（例如 537）。每条 Plasma 链都有一个操作者（这可能是一个中心化的角色，或者由多签控制，也可以是更复杂的东西，例如一条 PoS 链或 DPoS 链）。每隔一段时间（可以是 15 秒，也可以是一小时，或者介于两者之间的任何时长），操作者就会生成一个

“批处理”（batch），包含这段时间内所收到的所有 Plasma 链的交易。操作者生成一棵 Merkle 树，在索引为X的叶子节点处，如果资产 ID 为X的资产在这一批次中发生了交易，则叶子节点上存有对应交易，否则该叶子节点为零。操作者将这棵树的 Merkle 根发布到主链上。操作者还需要将每个索引X的 Merkle 分支发送给该资产的当前所有者。如果需要将资产从 Plasma 链提取至主链上，用户需要给主链的智能合约发送该资产最近一次交易对应的 Merkle 分支。智能合约随即开始了一个挑战期（例如7天），在此期间，任何人都可以尝试使用其他 Merkle 分支来证明：（i）用户在提取资产时并不拥有该资产，或者（ii）用户在某个时间点将资产发送给了其他人，从而使退出申请无效。如果在 7 天内没有人证明退出是欺诈性的，用户便可以成功取回资产。

Plasma 提供了比状态通道更强的功能：你可以将资产发送给从未加入二层的用户，锁定的资金也低得多。但这是有代价的：状态通道在 “正常运行” 时不需要将任何数据存入主链，但 Plasma 要求每条链每隔一段时间在主链发布一次哈希值。此外，Plasma 链中的交易没有即时性，即必须等待一批交易（也可称为 Plasma 区块）的证据（即那个 Merkle 根值）发到主链上。

此外，Plasma 和状态通道都有一个重要缺陷：其安全模型所对应的博弈论依赖于这样的想法：两个系统所控制的资产在逻辑层面都要有 "所有者"。只要资产所有者在乎自己的资产，那么当涉及该资产的状态变更 “无效” 时，资产所有者就会想办法出示变更 “无效” 的证明。这对一些应用来说无关紧要，但对不少其他应用来说这是个问题（例如 Uniswap）（译者注：对于 Uniswap 这样的 DEX 交易应用来说，其智能合约不属于任何人，与上述安全假设依赖的想法矛盾）。即使系统中一个对象的状态可以在未经系统所有者同意的情况下被改变（例如在基于账户的系统中，可以在未经所有者同意的情况下 增加其余额），也不能很好地与 Plasma 兼容。这意味着，在现实中部署 plasma 或状态通道时，都需要推演 “特定应用” 的潜在逻辑设计定制化方案，不太可能做出一个能完整模拟以太坊运行环境（即 “EVM” ）的 plasma 或状态通道系统。接下来让我们看看 rollup 是如何解决这个问题的。

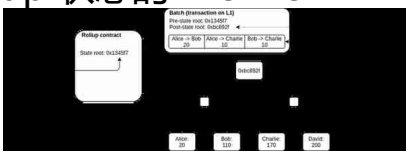
Rollup

另请参阅：EthHub 关于 optimistic rollup 的介绍，EthHub 关于 ZK rollup 的介绍Plasma 和状态通道方案都是 “完全的” layer 2 方案，因为它们将数据和计算都转移到链下（即自己的二层系统中）。然而，数据可得性的基本博弈论原理意味着这样的系统不可能安全地实现所有应用。Plasma 和状态通道通过明确资产对象和所有者之间的关系来解决这个问题，但这使它们无法完全通用。Rollup

则与前两者不同，是一种具有 “混合性质” 的二层方案。Rollup 将计算（和完整的状态存储）
转移到链下，但在链上保存了每笔交易的部分数据信息。为了提高效率，Rollup 使用了一系列花哨的压缩技巧，并尽可
能地用计算替代数据
。其结果是，这个系统的可扩展性仍然受限于底层区块链的数据带宽，但在此基础上实现的扩容倍数非常可观：在以太坊主链执行一笔 ERC-20 代币的转账大约消耗 45000 gas，但在 Rollup 中，每笔交易仅需要在主链上存储 16 字节数据，消耗的 gas 小于 300。数据存储在主链是 Rollup
的关键因素（请注意：这同将数据存储在 “IPFS” 上不同，因为 IPFS
不会对存储的数据进行共识，Rollup
的数据则必须
存储在区块链上）。将数据存储在主链并进行共识，使得任何人可以在本地处理 Rollup 中的所有操作，包括欺诈检测，发起提款，生成批处理等。因为不存在数据可得性问题，所以运营者如果作恶或者离线所造成的损失相对更少（比如他们无法造成长达 1 周的退出延迟），也在谁有权发布批处理这个问题上提供了更多的可能性，同时也使 Rollup 更易于理解。更为重要的是，不存在数据可得性问题意味着，资产无需和所有者有明确的逻辑映射关系。相比其他二层扩容方案，这是以太坊社区对 Rollup 感到更加激动的重要因素：Rollup 是具有通用性的，比如可以在 Rollup 中运行 EVM，从而使现有的以太坊应用可以在不写新代码的情况下迁移至 Rollup。

Rollup 究竟如何工作？

在主链上有一个智能合约，存有一个状态根—— 表征 Rollup 状态的 Merkle 根。（状态内容包括 Rollup 中的账户余额，合约代码等）



为了支持存款（译者注：将资产从主链的其他地方存入 Rollup
合约）和取款（译者注：解除 Rollup 合约对自己资产的控制，使之返回到链上其他地方），批处理中的事务的输入或输出可以是 Rollup
“外部”（译者注：即主链其他地方）的状态。如果批处理中有交易的输入来自 Rollup 之外，那么这个批处理操作会将主链其他地方的资产转移至 Rollup
合约中。如果批处理中有交易的输出来自 Rollup
之外，那么这个批处理会触发智能合约中的取款操作，将资产从 Rollup
取回主链。

整个过程就这么简单！不过还有一个细

节。
如何知道批处理执行完成之后的
状态根是正确的？
如果有人可以提交一个伪造的状态根，而不产生任何后果，他们就可以把所有的资产转移给自己。这个问题很关键，因为这个问题有两种截然不同的解决办法，从而形成了两种类型的 Rollup。

Optimistic rollups vs ZK rollups

Rollup 的两种类型包括：

1. Optimistic Rollup，使用欺诈证明解决上述问题：主链的 Rollup 合约记录了该 Rollup 内部状态根变更的完整记录，以及每个（触发状态根变更的）批处理的哈希值。如果有人发现某个批处理对应的新状态根是错误的，他们可以在主链上发布一个证明，证明该批处理生成的新状态根是错误的。合约校验该证明，如果校验通过则对该批处理之后的所有批处理交易全部回滚。
2. ZK Rollup, 使用有效性证明解决上述问题: 每个批处理中包含一个称为 ZK-SNARK 的密码学证明（例如使用PLONK协议，
证明新状态可以由旧状态经此批处理操作后转换而来（译者注：在 ZK Rollup 中，提交的密码学证明只需要具有尺寸小，验证快的特性即可，无需具有零知识性，所以只需要 SNARK 即可实现 ZK Rollup，不是一定需要 ZK-SNARK。事实上，在 ZK Rollup 中，因为批处理交易信息存储在主链上，而这些信息恰是密码学证明系统中所证明的内容，因此 ZK Rollup 本身并不具有零知识性，具有零知识性的 ZK Rollup 被称为 ZK ZK Rollup ）。无论批处理的计算量多大，都可以在主链上高效对证明进行验证。

这两种 Rollup 之间的设计权衡比较复杂

性质	Optimistic Rollup	ZK Rollup
每个批处理在主链上的固定 gas 开销	约 40,000（开销相对轻量，主要操作是改变状态根的值）	约 500,000（对于 ZK-SNARK 的验证开销相对较大）
提现周期	约 1 周（需要留出时间，看是否有人提出欺诈证明，从而取消无效提现申请）	非常快（在提交提现申请后的下个批处理即可完成）
技术复杂度	低	高（ZK-SNARKs 技术较新，在数学层面相对更复杂）

性质	Optimistic Rollup	ZK Rollup
实现通用性的难度	相对更容易（很快就可以将实现通用性 EVM 的 Rollup 部署在主网中）	相对更难（使用 ZK-SNARK 证明具有通用性的 EVM 执行操作比证明简单的计算操作难得多，当然（例如 Cairo）等工作正在努力简化该过程）
Rollup 中每笔事务的成本	相对更高	相对更低（如果事务数据中的某些部分只用来验证该事务的有效性，而不涉及状态变更，这部分数据就可以不存储在主链上，而在 Optimisitc Rollup 中需要存储所有数据，用于后续欺诈证明的生成和校验）
链下计算的开销	相对更低（虽然需要更多的全节点进行重计算）	相对更高（生成 ZK-SNARK 证明，特别是生成通用计算的证明开销可能很大，比直接计算要大好几千倍）

总的来说我自己的观点是，短期内 Optimisitc Rollup 很可能在实现具有通用性的 EVM 中胜出，ZK Rollup
很可能在简单的支付、转账和其他特定应用中胜出。但从中长期来看，随着 ZK-SNARK 技术的改进，ZK Rollup 将在所有应用场景中胜出。

剖析欺诈证明

Optimistic Rollup
的安全性取决于这样的想法：如果有人将一个无效的批处理发布到 Rollup 中，任何
同步了该链的人只要发现欺诈行为便可以向主链上的合约发布欺诈证明，证明该批处理是无效的，应该被回滚（译者注：不清楚作者这里的 “该链（the chain）”到底是指主链，还是指 Rollup，虽然都是对的）。