单次/Gas Tank/PostPayCard/Super paymaster v0.12

Gas Sponsor的产品形式

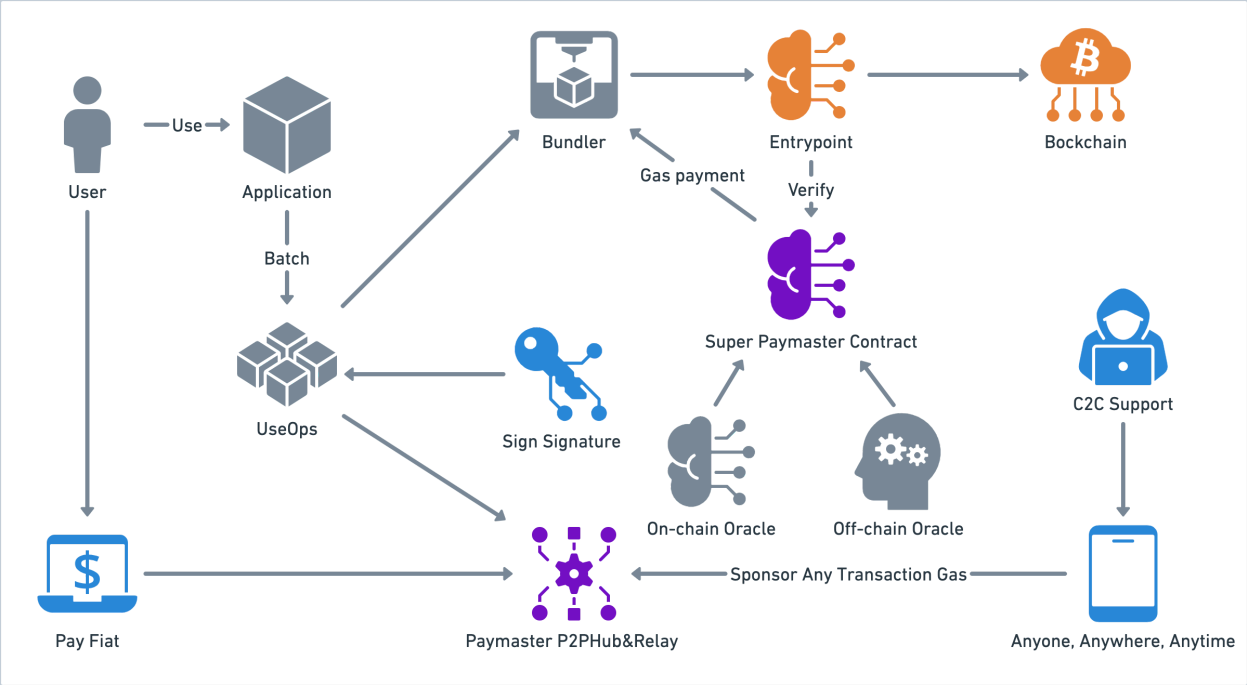
单次

是最基础的gas sponsor方式：支持多种协议的gas sponsor

1. 支持4337标准,对合约账户的gas sponsor
2. 支持3074方式下的AUTHCALL的交易方式的gas sponsor
3. 支持未来7560和其他链的Native Account Abstraction的的交易类型gas sponsor
4. 支持未来7560下的EOA的gas sponsor

5. TYPE1到TYPE4，所有账户类型，交易类型，当下和未来协议，我们都持续跟进和支持

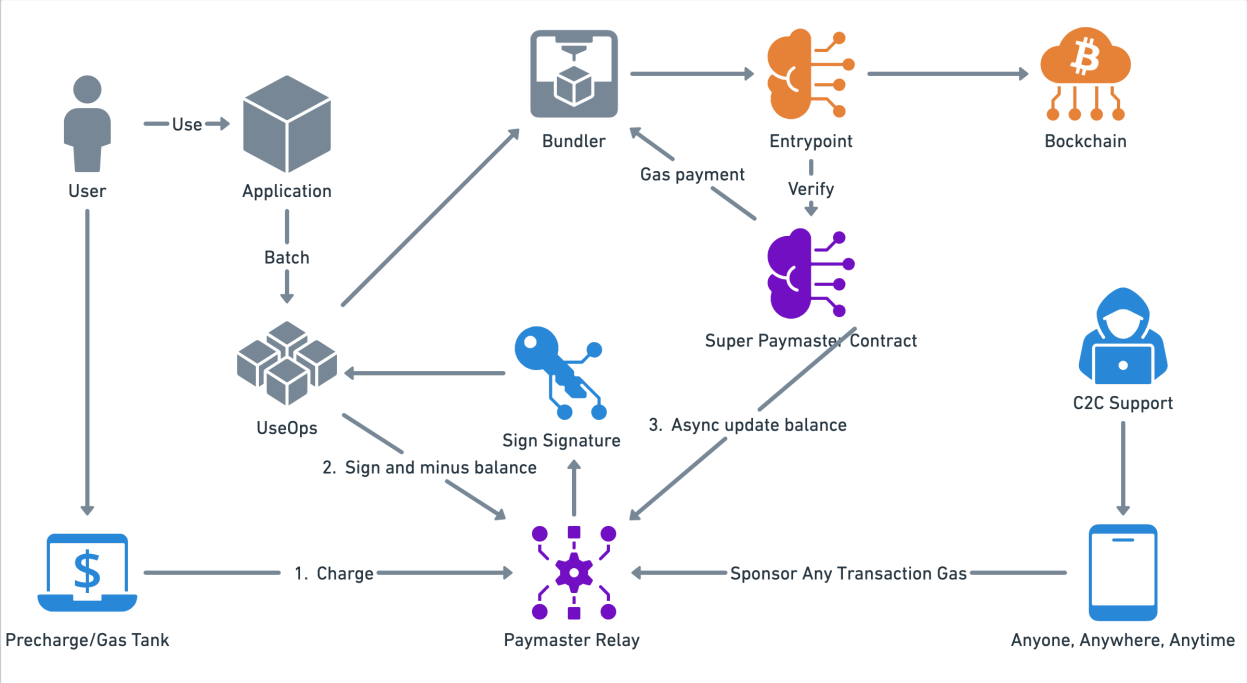
流程图：（4337版本，后面增加其他流程）



Gas Tank

1. 在单次基础上，改进用户支付的方式，从实时付给，改为了储值预付费和扣减账户余额，因此叫Gas Tank。
2. 整体流程在上述基础上改进如下：
    a. 依然需要找找Relay签名，但签名过程改进为依据用户NFT在Relay中心化账本记录下的余额扣减，然后签名，其他过程不变。
    b. 如果余额不足，则拒绝签名，返回余额不足错误。
    c. postOp后需要根据实际发生后的多余gas，要增加到用户合约账户？或者退回到中心化账户增加？这个反而可以伪实时（异步），实现方式可以是Event订阅后的update？

流程图



核心业务动作

1. Charge
    a. 建立一个NFT销售小的合约+前端（ENS项目在做），使用任何ERC20都可以购买；法币（信用卡）沟通后续也开通？
    b. 如果有便宜的平台/API，也可以用，但担心我们微利的手续费，例如OpenSea或者Rarely啥的
    c. 在relay自己后台建立account：NFT地址，充值ETH额度（如果有其他native token则增加列，例如StarkNet的native token）
    d. 以上操作会基于某个便宜的Layer2，目前首选OP（Arb)

2. Minus and sign（Deduction）
    a. 在估计UserOps的gas后，直接进行Account余额扣减，然后签名
    b. 余额不足要报错

c. 余额这里未来会涉及分润，没想好放哪里，就是串货：别人接单了但没有这个链，你来接，28分成。有点像mempool的逻辑。

3. Update
    a. 扣减的gas和实际花费的gas在postOp环节进行计算，有多余则发个event，让订阅的进行balance增加？或者其他方式？

PostPayCard（后付卡）

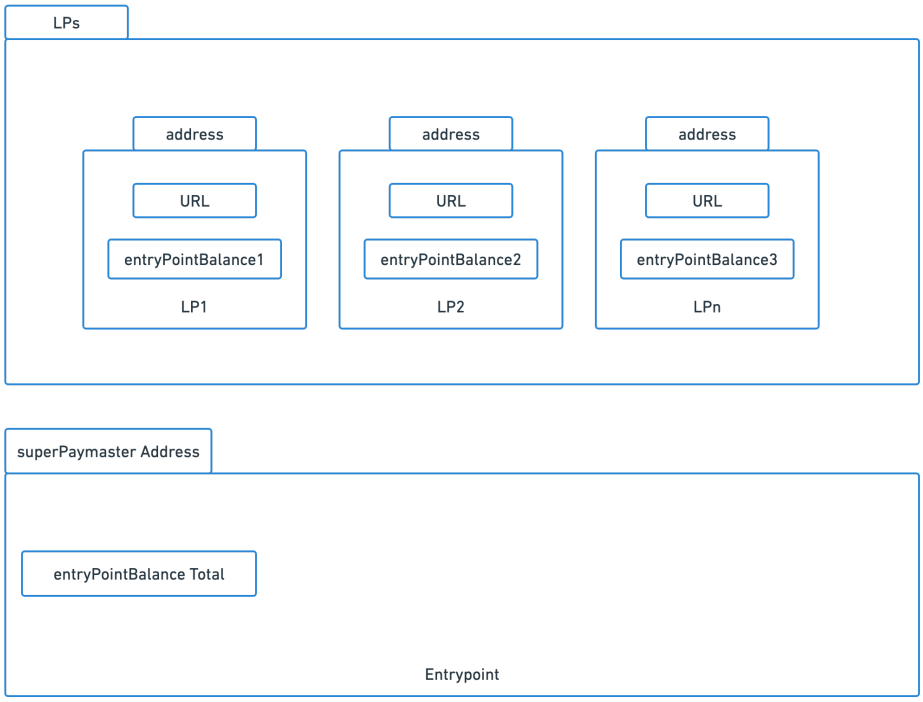
1. 在Gas Tank基础上改进
    a. 设立用户Gas模型：和常规交易合约，以及频次限制，甚至发起IP限制？
    b. 由社区发起，用户要stake一部分ETH，参与社区的card发行（后面设计模型）
    c. 用户和社区都会有收益，然后stake+收益来做月度结算，来扣减月度的gas sponsor费用
    d. 社区可以和ETHPaymaster合作，具备各种free gas event，PostPayCard优先自动获取进入白名单，其他人要设置？
    e. 在Gas Tank基础上，进行，设立reputation的评级机制，越高自由度越大

2. TODO

Super Paymaster Account model和Relay储值管理

几个用户使用模型和我们的Super Paymaster的模型以及储值（在Entry Point中）管理有关，这里也一并加上。

结构图



概念

_MIN_ENTRY_POINT_BALANCE

LPData storage lp = LPs[LPAddress];

lp.url = url;
lp.entryPointBalance

安全问题

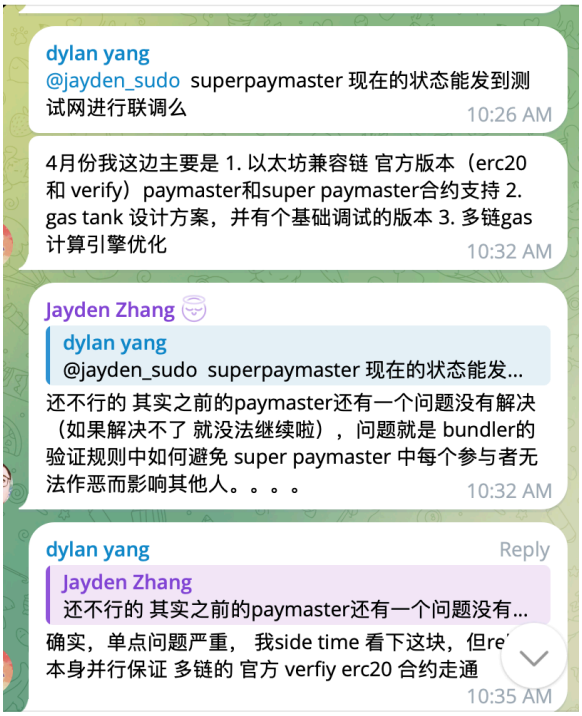
简单说就是上述模型bundler无法知道每个LP的存储额度，也就是上限是多少，只知道SuperPaymaster的存储额度entryPointBalance，因此有一个风险存在。

就是在单词每个LP提交自己的userOps的时候，需要的gas超过了LP存储在SuperPaymaster的gas额度，如果没有超过SuperPaymaster的额度，有点像透支，可以记账解决，但如果LP恶意的超过SuperPaymaster的额度，造成整个userOps revert，降低了SuperPaymaster的reputation。

relay可以带着LP的余额进行验证，超过余额就不签名，进入不了bundler，是不是就没有这个风险了？

解决方案：

1. 提高EP的ETH存储额度？恶意攻击依然可以消耗完，然后阻止后续userop提交，服务失效，不可行
2. Bunlder设法获得LPs的不同上限，然后做限制，需要自己建立Bunlder，或者提交一个新版本的Bundler标准，增加LPs的额度验证。
3. LPs增加一个链上Superpaymaster合约验证余额的动作签名，Bundler是否不改变的情况下，验证多一个签名验证？
4. LPs有自己的独立合约地址，Superpaymaster作为总的代付地址，存储上LPs的各个合约地址。或者bundler改逻辑，或者通过Superpaymaster路由来获得LPs的地址。
5. Jaydon：先做单个的Paymaster，然后再看？



安全问题再讨论

关于安全问题的集合讨论

背景

在讨论合约和安全的时候，Jaydon分析到了一个问题：简单说bundler没有LPs的各自余额信息，只有Superpaymaster的余额，这就可能造成攻击或者非恶意的超花：任意独立LP的stake余额是小于整体余额的，但他可以利用bundler不知道自己的余额信息差，利用bundler只能校验Superpaymaster余额，某恶意LP大量提交交易来独占bundler额度，而超出整体Superpaymaster额度后，bundler会扔掉本来应该合理进入bundler的其他LP的交易，造成交易失败。

性质判断

1. 首先bundler不是永远存在，4337如果有了3704+7560（bundler会淡化）或者Native Account Abstraction；未来bundler角色会消失（在native就不存在4337），直接和sponsor合约交互扣款，gas由paymaster提供，block builder审核（等于合并了bundler的功能）。
2. 其次在bundler存在的阶段，我们判断，大约是2年左右，不会超过三年，乐观甚至更快；在此阶段，所有的4337类别的交易，会由bundler来打包。
3. 而3704目前依赖AA原有路径实现gas sponsor，这次着急年底升级上线原来被挂起的3704，估计是大家对UX已经无法忍受了（主要说gas payment），而4337复杂的流程，高昂的合约部署成本（现在L2 gasfee降低了，但合约钱包依然高于普通EOA很多,4844后还没对比过？）也是选择3074上来试试的原因？
4. 补充，3074被7702,这个更兼容4337和AA roadmap的EIP替代了

动机

我们提出Superpaymaster合约的动机：

1. 单一地址
    a. 跨链一致的一个paymaster地址，给所有开发者以友好的开发体验（加上简单的SDK调用和多种产品方式支持）
2. 统一stake
    a. 所有LP（独立sponsor）都stake到superpaymaster合约内，整体做为一个paymaster服务，信用整体上升，成为大多数bundler、钱包和项目方的首选

b. 而基于统一的stake和sponsor流程，我们构建了整个产品流程（单次、储值gastank、后付费等提升客户体验的产品形式）和协作（去中心的Runner+LP+project）分润模式。

c. 这也是ETHPaymaster能够作为一个protocol 社区，独立可持续运营的原因：通过技术立足（单一地址和统一stake解决方案），贡献生态价值（weak censorship和开源、一键部署带来的多样性的paymaster），构建协作和可持续（token机制下的公共物品可持续）

3. 其他特性

a. 例如无成本跨链、支持LERC20、PERC20，gas tank等等，都是独立paymaster具备的特征，不是superpaymaster的核心特征，当然是具备价值的。

困难和解决方案

目前这个风险是存在的，核心是bundler的信息差

解决方案大概有几个思路（并非纯粹针对这个困难，而是从团队角度出发）

1. 当下的跨链和单次、gastank和后付费，继续设计开发，因为无论是super还是单个paymaster，都会有这个诉求，需要这部分组件。
2. 申请了ETHPaymaster.eth域名（昨天晚上开完会想了半小时，又起床申请的，paymaster.eth被20年就申请了，不知道咋购买？），作为退一步的解决方案。

a. 不同链有不同的子域名：op.ethpaymaster.eth, arb.ethpaymaster.eth, etc，映射到对应的合约地址，语义化比记录地址更简单友好，又和产品名字绑定，简单易懂；这个方案无论我们运行普通pyamster的多链，还是运行super paymaster，都有帮助。

b. 如果superpaymaster，则需要三级域名：Jason.op.ethpaymaster.eth，代表某个LP运行的paymaster relay，当然，前提是解决了bundler问题。

3. 提供Bunlder解决方案

a. 首先运行我们自己的paymaster，提供ens调用、跨链等等服务

b. 其他bundler如果调用我们superpaymaster，则需要改进

i. 常规是验证superpaymaster在entrypoint余额，改进则增加验证LP在superpaymaster的余额

1. 增加LP余额的获取
    a. 简单是通过我们superpaymaster合约提供余额查询
    b. 另外LP如果有三级域名，是否可以使用我们半中心化的服务，更新ENS中携带的数据？**需要实验**，类似不同域名的不同服务，例如传统 DNS TXT 记录的两个最重要用途是防止垃圾邮件和域名所有权验证，尽管 TXT 记录最初并非为这些用途而设计。
2. 增加验证，超出则抛弃userop，不超则正常执行

ii. 我们提供demo和SDK，快速搞定，获得了Superpayamster跨链的服务，如果必要，提供一定激励。

c. 提高EP的ETH存储额度？
    i. 恶意攻击依然可以消耗完，然后阻止后续userop提交，服务失效，不可行
    ii. 但我们的super合约验证签名有效性环节，可以增加余额冗余量判断，如果接近80%，根据日tx流量预测，你很快要超了，则发event到dashboard给LP，如果不充值，则降权（给更少的流量导入），引导用户在消耗完之前充值；通过平均数大体预测，会降低一点点可能性？

d. 需要自己建立Bunlder
    i. 自己建立一个符合上面bundler改进版的，肯定兼容
    ii. 或者提交一个新版本的Bundler标准，增加LPs的额度验证。不知道可能性是多少，为了统一地址和众包paymaster的特性

e. 其他环节卡一下
    i. 类似于网站流量的概念，我们是否可以构建流量先进入Rank高的机制，让bundler，钱包来选择。
    ii. 增加Rank可以采用增加额外stake（非entrypoin余额增加），提示reputation，也降低了恶意攻击的可能性。高rank的就是推荐，低rank就降流？
    iii. LPs增加一个链上Superpaymaster合约验证余额的动作签名，Bundler是否不改变的情况下，验证多一个签名验证？这个前面说了

f. 只做路由，不做stake

i. 这个有ENS方案可以替代，基本就是一个hao123的路由，没有联合众包的superpaymaster的集中stake，没有后续产品？

ii. LPs有自己的独立合约地址，Superpaymaster作为总的代付地址，存储上LPs的各个合约地址。或者bundler改逻辑，或者通过Superpaymaster路由来获得LPs的地址。

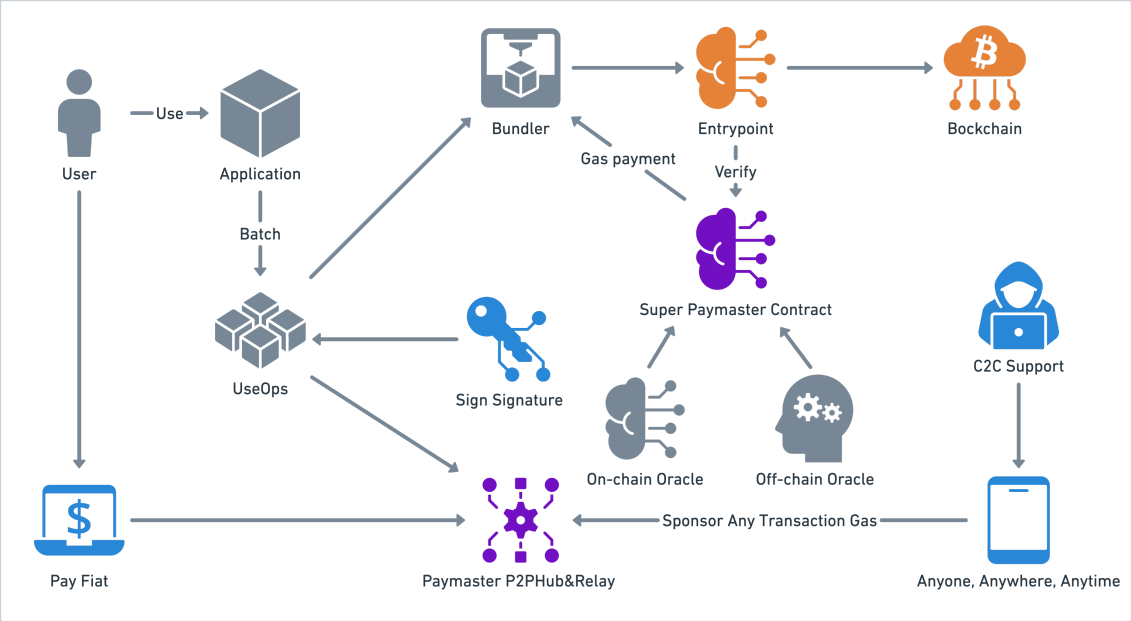
g. Jaydon：先做单个的Paymaster，然后再看？上面说了，哈哈，保守治理方案先做起来

h. **补充e的链上合约验证方案（23rd June）**

i. 因为superpaymaster合约扮演了传统paymaster合约，则需要对独立paymaster relay行签名后的（手续加入super要注册各子的公钥）验签；

ii. 验签环节，就需要对交易数据的gas进行计算后，扣减该relay在super的储值（参考上面的账户存储结构），理论上，如果储值不够，可以验签不通过，则避免了bundler无法获得具体某个relay的gas sponsor是否超出了自身储值上限造成的挤占其他paymaster交易的安全问题。

iii. 参考下面的结构，另外这个行为增加了对账户的gas计算（relay提供？）和对具体relay的储值余额的读取、扣减的操作成本。需要评估可行性、 成本和安全性是否满足预期。



结论

1. 先做自己的paymaster，基于跨链+gastank+增加ENS和3074（invoker+paymaster，方案待定）
2. superpaymaster低优先级构建，然后沟通一些bundler看反馈，同时自己构建bundler（Q3自己建立bunlder）

Reputation计算机制和风险

这个是bundler针对paymaster设立的reuptation，遵循7253（派生自4337协议）

TODO