Credibly Neutral Preconfirmation Collateral

by mteam from Spire Labs

inspired+reviewed by Justin Drake.

reviewed by Ladislaus, Murat, and others.

.

.

tl;dr

We introduce a design for a credibly neutral preconfirmations registry that solves the collateral problem for solo-stakers who want to earn additional yield from preconfs, while simultaneously improving capital efficiency for large operators who would like to opt-in to preconfirmations.

.

.

Introduction

There is a fundamental requirement of any proposer-commitment design that the L1 proposer must post some collateral. This is true for Based Preconfirmations (even with delegation) and other systems. This collateral is necessary because, according to the Ethereum protocol, a L1 proposer always has the option to self-build a block. The block that they build may violate commitments that they have made (or that other delegated entities have made).

In the next few paragraphs, we introduce a simple system for proposer to post this collateral in ETH (the most credibly neutral collateral). This system also enables large operators and solo-stakers alike to post collateral in efficient ways.

See my presentation introducing this topic for another intro.

Onchain Construction

The onchain construction is very simple:

We introduce the PreconfirmationRegistry

L1 smart contract that serves as a source of collateral for backing preconfirmations.

The contract provides 4 functionalities:

Penalties

The PreconfirmationRegistry

facilitates penalty application (i.e. slashing, freezing).

Delegates can choose what slashing conditions to opt-in to. These slashing conditions can be described using an EVM bytecode function.

Registration

The PreconfirmationRegistry

provides a way for a registrant to post ETH as collateral. A registrant can be any entity, it is represented in the registry as an Ethereum address.

Let a registrant’s collateral balance

= ETH deposited - ETH currently penalized - ETH withdrawn.

Delegation

Allows registrants to delegate their collateral to proposers.

Can be used to delegate to any proposer.

Let a proposer's collateral balance

be the sum of the registrant collateral balance of all registrants that have delegated to them.

If a proposer's collateral balance is above a parameter, MINIMUM_COLLATERAL

, they are a preconfer. If a proposer's collateral balance falls below MINIMUM_COLLATERAL

, they are no longer considered a preconfer.

When a penalty is applied to a preconfer, the penalty is applied to the registrant collateral of any registrants who delegate to the offending preconfer. The penalty is split proportionally amongst the collateral that is delegated to a proposer.

Note: Delegation can be done offchain and validated onchain in the case of a slashing event. Offchain signatures need to be shared publicly so that others (gateways) can verify that a given proposer is a preconfer.

Leader Election

The PrceonfirmationRegistry

also elects a leader for the current L1 slot. It does this by checking the lookahead to determine the next proposer in line who is a preconfer (meets the MINIMUM_COLLATERAL

threshold.) Based Rollups can select this elected leader as the sequencer for their chains to achieve composability through shared sequencing.

We call this leader the registry sponsored leader

. Any rollup interested in composability can use this leader. Rollups that have other requirements for their leader (regulatory, financial) can use their own leader election system, but may ise the registry sponsored leader as a starting point.

[

image

1658×948 110 KB

](https://ethresear.ch/uploads/default/original/3X/d/3/d3d476034f8a84825dfc2b477c02d88c53eef94f.png)

Offchain Construction

Underwriters

The most important offchain construction that we introduce is the Underwriter. The Underwriter acts as a registrant and posts at least MINIMUM_COLLATERAL

ETH as collateral to the PreconfirmationRegistry

contract on behalf of a group of proposers. The Underwriter accepts collateral from proposers in any form (a liquid staking token, equity, reputation).

Underwriters could be trustless DeFi applications, offchain market makers, etc.

Proposer Pools

While large staking operators form large trust networks between many proposers and can provide collateral on their own, solo-stakers do not have the same access to resources.

Solo-stakers may form "Proposer Pools" of a collection of proposers, all restaking their ETH to a single registrant (through an underwriter) to reach a registrant balance of MINIMUM_COLLATERAL

or more.

Note: Proposer pools have a 1/n trust model: a single proposer could renege on preconfirmation promises for a significant profit but the entire pool would be penalized. To disincentivize this situation, proposer pools could be designed so that EL rewards are distributed between all proposers in the pool (similar to smoothing pools). With a MaxEB increase above MINIMUM_COLLATERAL

and DVT, the trust model becomes n/3.

[

image

1521×1672 280 KB

](https://ethresear.ch/uploads/default/original/3X/a/9/a9982df96fdbf69566a89c5cc1e4306b9bca08d4.png)

Considerations

- credible neutrality:

The PreconfirmationRegistry

only accepts the purest form of collateral: ETH. This avoids any dependence on any LST/LRT and minimizes smart contract risk.

- centralization:

The PreconfirmationRegistry

does not introduce any permissioned actors. Underwriter centralization is not dangerous to the protocol.

- collateral depeg risk:

An external collateral depeg will not affect the collateral that backs preconfers. Underwriters are only vulnerable to permanent depegs as they are not forced to sell during a temporary depeg event.

- smart contract risk:

The PreconfirmationRegistry

is a very simple contract but should be audited extensively. Underwriters are especially incentivized to audit the PreconfirmationRegistry

contract.

- roadmap compatibility:

Proposer Pools have even better trust models with a MaxEB increase like EIP-7251 and DVT. With ePBS, Execution Tickets, Execution Auctions / APS-Burn or any other upgrade where proposers do not serve as monopoly sequencers, the PreconfirmationRegistry

is no longer useful.

- mass exits:

To avoid a situation where a registrant withdraws collateral that is actively backing a preconfirmation, exits must be delayed. This mandates that there are three states that a proposer can exist in: [INCLUDER, EXITING, PRECONFER]

. New preconfirmations should not be backed by collateral from EXITING

proposers.

- forced ejection:

In a penalty event, some preconfers may no longer have enough unpenalized collateral to back their preconfirmations. If these preconfers are in the lookahead and actively providing preconfirmations, they must be immediately exited and treated as INCLUDER

proposers. Registrants who wish to avoid forced ejection of their delegated proposers should top up their registrant balance to a level significantly higher than MINIMUM_COLLATERAL

.

- Preconf Chaining:

Most preconfirmations are only backed by a single proposer (and therefore a single collateral balance), this enables registrants to delegate to many proposers simultaneously. There may be situations where preconfirmations are backed by multiple proposers (think conditional preconfirmations or multi-block preconfirmations). In these situations, the collateral that backs a preconfirmation is the combined unique

proposer collateral balance of proposers involved. The odds of preconf chains applying to proposers using the same collateral are higher for registrants with more delegates, and therefore registrants who can afford to post extra collateral.

Bonus: Dynamic Collateral Requirements

The MINIMUM_COLLATERAL

parameter is a key parameter for preconfirmation security and bootstrapping a robust Preconfer set. Setting MINIMUM_COLLATERAL

ex-ante for all preconfirmations may lead to inefficiencies. For example, many retail users don't need thousands of ETH backing their preconfirmations. Indeed, many types of transactions (ETH/ERC20 transfers) are time-insensitive; preconfirmations for these transactions do not require huge amounts of collateral.

The solution to this inefficiency is setting a low MINIMUM_COLLATERAL

parameter and allowing users to set their own collateral requirements (along the lines of User-Defined Penalties: Ensuring Honest Preconf Behavior by Jonah from Blockchain Capital).

Note that it is still necessary to set a MINIMUM_COLLATERAL

value that is significant (likely more than 32 ETH) to avoid an adverse selection problem: A user might not be able to get a preconfirmation with a high collateral backing if a low-collateral Preconfer is first in the lookahead, even if a high-collateral Preconfer is next. More research is required to determine this parameter.

Bonus: Slashing Conditions

Slashing conditions represent generic agreements between proposers and users. Delegates (proposers) can choose what slashing conditions to use. To declare their preferences, delegates can sign off on EVM bytecode that represents a function. When executed, this bytecode returns the penalties that should be applied:

struct Penalty { uint weiSlashed, uint weiFrozen, uint blocksFrozen, // other penalties // }

Rollups that want to incentivize proposers to adopt their slashing conditions can set up incentives (tokens, fees, etc.) out of the registry.

Note:

Registrants might also choose slashing conditions for their delegates. For example, an Underwriter might accept collateral from an operator and register with the slashing conditions that the operator desires. This simply a way for operators to manage the slashing conditions that they need to think about.

References + Resources

All documents used can be found here.

Especially useful:

- Justin's Based Rollup post
- Justin's Based Preconfirmations post
- Jonah's User-Defined Penalties… post
- Limechain's Based Preconfirmations Research repo

Similar (unrelated) work:

- Primev's Registry
- Puffer's Registration System
- Cairo's preconf-operator code
- various Smoothing Pool designs