This is a solution for Data availability problem

of Plasma EVM designed by Onther Inc.

And I believe this can solve most of the problems. I'm looking forward to get lots of feedback from you.

This is very simplified version. I highly recommend to read this full version.

A proposal for Data availability Solution of Plasma EVM

(Korean)A proposal for Data availability Solution of Plasma EVM

Abstract

This article proposes a model to address the most problematic Data Availability (DA) inPlasma EVM. This model has a new 'User Request Block'

, which is a way for ensuring vaild Exit for users in case of data withholding, while leaving the judgment on DA entirely to the individual user. It also introduces a dynamic fee model

to prevent infinite loop attacks by malicious users pretending to behave as if there were DA problems.

New Exit model : User Request Block

Glossary

Non-Request Block(NRB)

: Same as nonRequestBlock in Plasma EVM

User Request Block(URB)

: Request Block submitted by a user. Unlike the existing Request Block, it contains only transactions that reflect the Exit Request for URB of submitter or other user.

Operator Request Block(ORB)

: Request Block submitted by the Operator. It is same as the requestBlock in Plasma EVM.

Exit Request for ORB(ERO)

: Exit Request using ORB

Exit Request for URB(ERU)

: Exit Request using URB

Rebase

: If the URB is submitted based on the most recent finalized block, all child blocks that are submitted but not finalized will be located behind the corresponding URB and transactions that conflict with the URB will be reverted. This is called a Rebase

.

Confirmation does not exist anymore in the new model. In addition, whether DA problem occurs or not, is not judged at all in the process of mining and submitting the block. Instead, users who noticed that there was DA problem in child chain can safely exit by committing an URB based on the most recent defined block including their own and other user's ERUs. Once the URB is committed, then the operator should Rebase

the unfinalized blocks to reflect the contents of the URB. If it is judged that there are no problems with the user's perspective, users can wait until the operator includes one's Exit request, using ERO instead of ERU like the previous model.

(If you want to fully understand the Rebase

, please check this.)

Fee model against Infinite loop attack

Infinite loop attack

The new model left the judgment of ‘block withholding’ to the individual users, and in case of a problem, URB can be submitted for safe Exit at any time. However, there is a fatal vulnerability issue in this model. It is a kind of infinite loop attack using Rebase

.

Fee model

We have introduced a model to charge fees for URB and ERU to prevent such attacks.

The design objectives of the fee model are as follows.

1. If the submission of URB is close to the probability that it is an attack by malicious users, the fee should be charged high. And if it is close to the probability that it is an escape from a problem, the fee should be set low.
2. The number of URB commits that generate Rebase should be as low as possible.

DA probability

If an independent individual makes one’s own judgment on DA matters, it is the individual’s judgment that is most relevant to the probability of DA problems. That is, the greater the number of users who believe that there is the DA problem, the greater the likelihood of DA occurrence. On the contrary, if there are fewer users who believe there is DA problem, chances are high that there were no problems with the Child Chain. Therefore, we will design a model that estimates the probability of a DA issue and adjusts the fees for the URB and ERU accordingly through user’s judgments about DA problems, i.e. the number of ERUs

.

Cost function

To satisfy the first principle, the higher the number of ERUs in the URB, the lower the URB’s submission costs and the cost of the ERB. In addition, to meet the second principle, it would be desirable to increase the extent of the decreasing cost as the number of ERUs increases.

C_{URB}

: Cost for submitting URB

C_{ERU}

: Cost for Exit by ERU

N_{ERU}

: The number of ERUs in URB

As defined above, a cost function meeting the above conditions may be like below.

Cost of submitting the URB

Cost of ERU