AIP-7: Arbitrum One Governance Parameter Fixes

Non-Constitutional

Abstract

Three independent issues have been identified in the Arbitrum One Governance system and the current proposal aims to address them. Given this is maintenance of the system, after the forum discussion period it will skip the Snapshot temperature check and go directly for an on-chain vote.

Specifications

1. Updating the airdrop distributor fee sweep address to the DAO Treasury's address

The Arbitrum DAO airdrop was distributed to users via theTokenDistributor contract. The recipients are able to claim their tokens until the Ethereum block #18208000

(estimated to be created on the 24th September 2023).

After the claim period is over, unclaimed leftover tokens may be swept over to the specified sweepReceiver address, which is currently set to the L2 treasury timelock. Per the Governance Architecture documentation, this should be set to theDAO Treasury's address.

Fix:

Call the the 'setSweepReciever()' function on the TokenDistributor contract, and include theDAO Treasury address as the _sweepReceiver(address).

2. Sequencer gas fee reimbursement parameterization

The Arbitrum One sequencer pays the necessary gas fees for posting user transactions to Ethereum. This is done through transactions to the Sequencer inbox.

The sequencer gets reimbursed for these fees in Arbitrum One. The reimbursement is calculated by ArbOS, but two parameters are currently incorrectly configured.

1. The Sequencer Inbox has a fixed cost associated with including a transaction. The value is currently configured to 100000 Ethereum L1 gas units - this can be viewed in the ArbGasInfo precompile through the getPerBatchGasCharge function. If you inspect transactions live in the system (sample from June 15th and July 17th), it is possible to see that the fixed cost of including a batch is actually much closer to 240000 Ethereum L1 gas units.

2. The ArbOS L1 pricing system features an optional "amortization cost cap" which is intended to subsidize the fixed posting cost for chains with low activity or AnyTrust chains whose fixed cost is much larger than other data posting costs. This feature was not intended to be enabled on Arbitrum One. As such, the cap was set to its maximum value, 2^64 - 1 (this can be viewed in the ArbGasInfo precompile through the getAmortizedCostCapBips function). However, this did not fully disable the cap. As visible in the code, the amortization cap is only disabled with a value of 0. With the cap set to its maximum value, but still enabled, the cap would prevent the L1 pricer from taking into account the cost of multiple batches posted in the same L1 block. That's because it would consider the costs of all but the first batch as having a weight of zero, because no time passes since the previous batch if the batches are in the same L1 block. Setting the amortization cap to 0 fixes this issue by bypassing the previously linked if statement to fully disable the amortization code.

The combination of these two issues add up to gas funds being incorrectly charged to end users, thus not fully reimbursing sequencer operations. This fix will increase fees for users but they will now reflect the actual costs of the system - it is expected to be a minor difference.

Fix

Transaction to ArbOwner precompile calling setPerBatchGasCharge(int64) with the intended value '240000'.

Transaction to ArbOwner precompile calling setAmortizedCostCapBips(uint64) with the intended value '0'.

3. L1 Core Governance Timelock scheduleBatch Bug

When executing a batch of operations on theL1ArbitrumTimelock, if more than one operation creates a retryable ticket (i.e., more than one operation targets an L2 chain), the full msg.value value will be forwarded to each one. If not properly constructed, this can lead to retryable tickets that fail to get created. While there are workarounds, the current implementation is error-prone and the fix allows for more graceful creation of several L1 to L2 operations.

Fix

Upgrade the implementation of the L1ArbitrumTimelock with the change here.

–

The fixes were implemented through Governance Action contracts and can be viewed on the Governance codebase. They have been audited by Trail of Bits and no issues were identified - the audit report will be shared publicly soon.

The Action contracts have been deployed to the following addresses and can be verified/audited by the community:

- UpdateGasChargeAction: UpdateGasChargeAction | Address 0x7b1247f443359d1447cf25e73380bc9b99f2628f | Arbiscan
- SetSweepReceiverAction: SetSweepReceiverAction | Address 0xbaba4daf5800b9746f58c724f05e03880850d578 | Arbiscan
- UpdateL1CoreTimelockAction: UpdateL1CoreTimelockAction | Address 0xbaba4daf5800b9746f58c724f05e03880850d578 | Etherscan