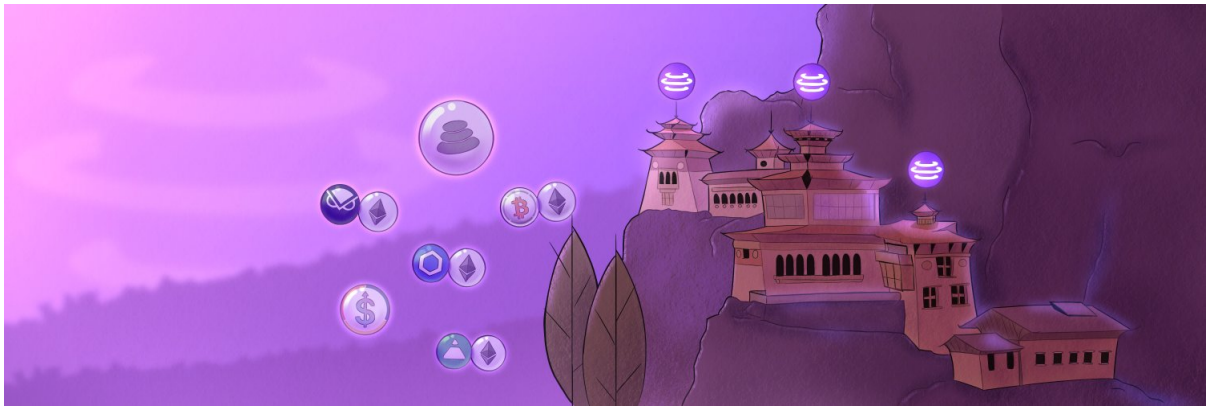

Aura governance upgrades report

1. Context

Aura Finance is a protocol built on top of Balancer to provide maximum incentives to Balancer liquidity providers through social aggregation of deposits and Aura's native token. Aura can be seen as a spin-off of Convex Finance, which implements a similar model for Curve.

One of the key components of this protocol is the `Booster` contract, the main deposit contract for liquidity pool tokens. Its main responsibility is to allow liquidity providers from different pools to stake their tokens all together in a single place, collect the corresponding rewards, and track the accountancy properly to redistribute these among all the participants.

2. Issues

In order to create new rewards for a pool, `addPool` must be called on the `Booster`. This method is only callable from the `PoolManagerProxy`, which at the end of the ownership chain is controlled by Aura's governance going through `PoolManagerSecondayProxy` and the `PoolManager`.

A potential situation was reported that involves a governance griefing vector. Aura's governance could freeze withdraws by using a specific flow that involves overflowing the queued rewards, shutting down a pool, and then re-adding a fake gauge that uses an out-of-order system shutdown to transfer gauge tokens out of the `VoterProxy`, making use of an specific method called `forceAddPool` on the `PoolManager`.

3. Approach

On one hand, Aura's contributors proposed two main changes to fix the reported issues: first removing `forceAddPool` on the `PoolManager` contract, and then provide a new

implementation of the `ExtraRewardsStash` contract in order to handle secondary reward tokens in a more robust manner to avoid potential overflows.

On the other hand, this opportunity was seized to reduce part of the governance functionality around the `BoosterOwner`. In particular a new wrapper contract was built on top of it to seal in the new `ExtraRewardsStash` and `PoolManager` implementations.

4. Fixes

All the contract changes can be found here, while the corresponding upgrade scripts and tests can be found here.

4.1. Pool manager upgrade

A new `PoolManager` implementation was introduced removing `forceAddPool`. As explained above it was one of the functionalities that could have been used maliciously by Aura's governance to manipulate pool withdrawals and user deposits.

Apart from that, a version of `addPool` that allowed parameterizing the stash version was removed too leaving only a version of `addPool` that forces the third version of the stash. A diff compared to the previous implementation can be found here.

4.2. Extra reward stash upgrade

A few changes were introduced to the `ExtraRewardStashV3` contract to make sure extra rewards' accountancy could not overflow and freeze reward pools. The main concept here is the `StashToken` which acts as a wrapper contract on top of the secondary reward tokens, allowing governance to decide if these should be processed.

4.3. Booster owner wrapper

A new wrapper contract called `BoosterOwnerSecondary` was introduced in order to proxy all functionality exposed by the `BoosterOwner` contract while adding specific control flows and validations. This new implementation cannot be changed, once the Aura's governance transfer the ownership of the `BoosterOwner` to `BoosterOwnerSecondary` it cannot be rolled back.

This new implementation carries a specific logic to seal the stash implementation. Once this method is called, the stash implementation cannot be changed anymore.

Facundo Spagnuolo

Ethereum developer and security researcher