title: [ARFC] Introducing “StrategicAssetManager” by Llama

author: Llama - @TokenLogic, @dydymoon and Fermin Carranza

discussions: [TEMP CHECK] Treasury Management - Introducing StrategicAssetManager

date: 2023.07.24

Summary

This publication progresses the StrategicAssetManager by Llama from [TEMP CHECK] to [ARFC].

The StrategicAssetManager is a dedicated contract for managing Aave DAO’s strategic assets, at launch it will manage the DAO’s veBAL holdings. StrategicAssetManager provides the functionality needed to maximise the DAO’s financial upside whilst minimising the DAO governance’s overhead when managing the assets.

Motivation

Please read the iniatial forum post. Building on this post, the below contains more details into how the contract will work.

To limit the surface area of the contract, an AllowList

will define which contracts can be intereacted with. This is important as the veBAL position requires onging and frequent attention. To avoid governance overload, the Guardian

role can be assigned to a multisig. The Guardian

acting as an Asset Manager

then has the ability, along with the ShortExecutor (aka Governance)

to interact with a subset of the Strategic Asset Manager contract’s functions. The AllowList

eliminates the risk of the AssetManager

interacting with a milicious contract, running away with the DAO’s funds or allocating veBoost or gauge votes to non-approved pools. The intent is to implement the highest heirarchy of control.

The AllowList

is defined and solely maintained by the Governance

. The AssetManger

role does not have the ability to add/remove contract addresses from the AllowList

. This eliminates the AssetManger

multisig signers from colluding and utilising Aave DAO’s assets for self gain. Aave’s Snapshot is to be used for directional changes/mandates in how the gauge votes and veBoost is to be used.

The below provides a high level overview of the functionality provided by the Strategic Asset Manager Contract

:

- Lock/relock B-80BAL-WETH
- Participate in gauge votes
- Delegate governance voting rights
- Allocate boost
- Delegate boost
- Allow boost on Warden
- Sell boost on Warden
- Claim protocol fees & other rewards

- Transfer assets
- Ability to lock other strategic assets
- Receive assets from the ER
- Receive assets from the collector
- Allocate rewards to create votes incentives on Quest
- Withdraw unspent votes incentives on Quest
- Claim votes incentives received by voting for Aave pools on Quest

The contract is upgradeable and controlled by Aave's [Short Executor

](https://docs.aave.com/developers/guides/governance-guide#the-short-executor-is-used-for-non-governance-related-proposals-such-as-asset-listing-parameter-upda) role via governance. Each function on the contract can be implemented via the ShortExecutor

whilst also having the ability to assign and remove, an AssetManager

role to a community elected address.

Some of the functionality and tokens the contract can interact will be available on v1 of the implementation, while others will be upgraded to include.

Specification

The Strategic Asset Manager will contain an AllowList

defining which contracts the contract can interact with. The AllowList

can only be updated by Aave Governance via the ShortExecutor

.

The below details the specific functions on the contract with a brief description on how each works. These functions are available on v1.

function addToAllowList(address token)

This function can only be called byShortExecutor

. Aave governance defines which contracts the StrategicAssetManager

contract can interact with.

function removeFromAllowList(address token)

external { Callable by governance }

This function can only be called byShortExecutor

. Aave governance can remove the ability of the contract to interact with a certain token.

function updateGuardian(address)

external { Callable by governance or guardian }

Callable by governance, only ShortExecutor

or the current Guardian

can assign the Guardian (AssetManager)

role to an address. Can be used to revoke the role from the assigned multi-sig.

function getExpectedAmountOut(fromToken, toToken)

external { Lets users view how much they can expect out at any given time }

Function shall query to determine the amount of token received from swap.

function swap(address fromToken, address toToken, address recipient)

external { Callable by governance or assetManager Validate fromToken is allowed Validate toToken is allowed Validate fromToken balance is > 0 for trade Recipient is this contract only }

Function shall approve and swap the specified asset/s via Cowswap using Milkman to protect against MEV attacks.

function cancelSwap(address tradeMilkman, address fromToken, address toToken, address recipient, uint256 amount)

external { Callable by governance or assetManager Validates token is allowed Reverses trade() }

Function cancels a specific swap.

function lockVeToken(address token)

external { Only callable by governance or assetManager Validates token is allowed Depsosits into Balancer's voterEscrow contract and locks for the specified time duration }

Function deposits B-80BAL-20WETH (or other token that can be locked for corresponding veToken) into voterEscrow contract with a defined lock period.

function lockSdToken(address token)

Same as above but for sdToken.

function voteForGaugeWeight(gauge address, uint256 amount)

external { Only callable by governance or assetManager Validates address is allowed Votes allocated amount of veBAL on gauge }

Function uses the veBAL holding to vote BAL emissions for a specific gauge.

function setDelegateSnapshot(address)

external { Only callable by governance or assetManager Validates address is allowed Delegates veBAL governance rights to address via Snapshot for balancer.eth }

Function delegates veBAL vote rights on Balancer DAO Snapshot (balancer.eth) to an AllowList

address.

function voteAragon(address)

Function allows contract to vote on Aragon. veCRV uses Aragon for voting while veBAL uses Snapshot.

function sellBoost(address)

external { Only callable by governance or assetManager Validates address is allowed Delegates boost to specified address }

Sells contract’s Boost for a specified period of time.

function updateBoostOffer(address)

external { Only callable by governance or assetManager Validates address is allowed Delegates boost to specified address }

Function updates the offer to sell boost.

function removeBoostOffer(address)

external { Only callable by governance or assetManager Validates address is allowed Delegates boost to specified address }

Function removed the offer to sell boost.

function buyBoost(address)

external { Only callable by governance or assetManager Validates address is allowed Sells boost to specified address }

Function purchases boost in order to boost expected emissions.

function claimRewards(address)

external { Only callable by governance or assetManager Validates address is allowed Claims rewards from veBAL }

Function claims rewards from selling boost.

function withdrawERC20(address token, receiving address, uint amount)

external { Only callable by governance Validates token and address is allowed Transfer token to specified address }

Function transfers assets held in StrategicAssetManager to a specified address. Only callable by governance.

Further information about the deployment can be found here.

Copyright