

Third[Guard]



ENS DAO <> Karpatkey: Policy Audit

Date	21 August 2024
Author	ThirdGuard
Website	https://thirdguard.com
Contact	pri@thirdguard.com , jono@thirdguard.com
Current Policy	Roles: 0x703806e61847984346d2d7ddd853049627e50a40 - MANAGER
New Policy Request	C5Twf3khKv2Ny8PvzoARgHFKFFK8vliiNR7nDkrlM



Findings

Across both the new functions audit & modifications audit we are pleased to report that we have found no material findings. Policy changes requested are considered bonafide actions needed by the manager to carry out their DeFi operations.

Findings Classification

Finding	Symbol
Informational	
Warning	
Critical	

New Functions Findings

No.	Contract	Function	Parameter	Category	Finding
1	Balancer osETH/wETH StablePool Aura Deposit Vault	getReward (address _account, bool _claimExtras)	_account		In this function rewards cannot be redirected, they are both claimed for and sent to _account meaning scoping this variable poses no extra risk, but is unnecessary.
2	CowSwap Order Signer	unsignOrder	order.sellAmount & order.buyAmount		Unconstrained access to these parameters allows a manager to control the minimum acceptable amount of tokens out from a swap. A manager could attempt to grief this by submitting swaps with excessively high amounts of implied slippage. CowSwap solvers should in theory compete to execute trades at much more favourable rates of slippage than the implied maximum the manager submits, but as the behaviour of said solvers is not guaranteed, the DAO should be aware of this grieving vector. See 5.1 for more.

Modifications Findings

No Findings

Findings Resolutions

No.	Finding
2.	Karpatkey commented that the scoping of these parameters will be challenging and impractical due to the requirement of scoping relative to an oracle. In light of this, Karpatkey has taken the position that they have outsourced sound trade execution to the solvers of CowSwap to ensure trades are executed in a competitive fashion at minimum levels of slippage.

New Functions Audit

1. Stakewise v3 Eth Genesis Vault

[0xAC0F906E433d58FA868F936E8A43230473652885](#)

Description

Initiates the staking process, manages staking, provides staking rewards, and handles migrations and upgrades.

1.1 Function

```
function updateStateAndDeposit(  
    address receiver,  
    address referrer,  
    IKeeperRewards.HarvestParams calldata harvestParams  
) external payable returns (uint256 shares)
```

Description

Updates the vault state, deposits ETH, and mints shares to the receiver. It also emits referral addresses and handles harvests vault rewards.

Parameter Sensitivity

HIGH

- receiver - The address that will receive the minted vault shares. Critical as it determines who gets ownership of the newly minted shares.

LOW

- referrer - The address of the referrer. No known active referral program is live, zero address currently passed to ignore referrals. Manager could pass 3rd party address here once a program is live, but the risk to client remains low given the assumption that referral rewards will be trivial or non-existent.

LOW

- harvestParams - Parameters needed to update vault state, specifically used for keeper harvesting rewards. Typically retrieved from Stakewise subgraph or latest rewardsUpdated event from the relevant keeper contract.
 - harvestParams.rewardsRoot - The rewards merkle root.
 - harvestParams.reward - The Vault cumulative reward earned since the start. Can be negative in case of penalty/slashing.
 - harvestParams.unlockedMevReward - The Vault cumulative execution reward that can be withdrawn from shared MEV escrow. Only used by shared MEV Vaults.
 - harvestParams.proof - Proof to verify that Vault's reward is correct.

Access Control Recommendations

Parameter	Permission	Operator	Value	Policy Complies
receiver	Scoped	equal	AVATAR	✓
referrer	Unscoped			✓
harvestParams	Unscoped			✓

Option	Operator	Value	Policy Complies
Enable sending ETH	equal	true	✓

1.2 Function

```
function mintOsToken(  
    address receiver,  
    uint256 osTokenShares,  
    address referrer  
) external returns (uint256 assets);
```

Description

Mints OsToken shares and returns the number of assets minted to the receiver.

Parameter Sensitivity

HIGH

- receiver - The address that will receive the minted osTokens. Critical as it determines who gets ownership of the tokens.

LOW

- osTokenShares - The amount of osToken shares to mint.

LOW

- referrer - The address of the referrer. No known active referral program is live, zero address currently passed to ignore referrals. Manager could pass 3rd party address here once a program is live, but the risk to client remains low given the assumption that referral rewards will be trivial or non-existent.

Access Control Recommendations

Parameter	Permission	Operator	Value	Policy Complies
receiver	Scoped	equal	AVATAR	✓
osTokenShares	Unscoped			✓
referrer	Unscoped			✓

1.3 Function

```
function burnOsToken(  
    uint128 osTokenShares  
) external override returns (uint256 assets)
```

Description

This function allows users to burn their osToken shares and receive the corresponding assets in return.

Parameter Sensitivity

LOW

- osTokenShares - The amount of osToken shares to be burned.

Access Control Recommendations

Parameter	Permission	Operator	Value	Policy Complies
osTokenShares	Unscoped			✓

1.4 Function

```
function enterExitQueue(  
    uint256 shares,  
    address receiver  
) external returns (uint256 positionTicket);
```

Description

This function allows users to enter the exit queue for withdrawing shares from the vault.

Parameter Sensitivity

LOW

- shares - The number of shares to be withdrawn. This parameter determines the amount of assets the user will receive upon exit.

HIGH

- receiver - The address that will receive the withdrawn assets. This is a critical parameter as it determines where the assets will be sent after the withdrawal process is complete.

Access Control Recommendations

Parameter	Permission	Operator	Value	Policy Complies
shares	Unscoped			✓
receiver	Scoped	equal	AVATAR	✓

1.5 Function

```
function claimExitedAssets(  
    uint256 positionTicket,  
    uint256 timestamp,  
    uint256 exitQueueIndex  
)
```

Description

This function allows users to claim assets that have completed the exit process from the vault.

Parameter Sensitivity

LOW

- positionTicket - Unique identifier for the exiting position.

LOW

- timestamp - Timestamp of when the exit was initiated.

LOW

- exitQueueIndex - Index in the exit queue for a position.

Access Control Recommendations

Parameter	Permission	Operator	Value	Policy Complies
positionTicket	Unscoped			✓
timestamp	Unscoped			✓
exitQueueIndex	Unscoped			✓

1.6 Function

```
function deposit(  
    address receiver,  
    address referrer  
) public payable virtual override returns (uint256 shares)
```

Description

This function allows users to deposit ETH into the vault and receive shares in return.

Parameter Sensitivity

HIGH

- receiver - The address that will receive the shares. This is a critical parameter as it determines who will own the shares minted from the deposit.

LOW

- referrer - The address of the referrer. No known active referral program is live, zero address currently passed to ignore referrals. Manager could pass 3rd party address here once a program is live, but the risk to client remains low given the assumption that referral rewards will be trivial or non-existent.

Access Control Recommendations

Parameter	Permission	Operator	Value	Policy Complies
receiver	Scoped	equal	AVATAR	✓
referrer	Unscoped			✓

Option	Operator	Value	Policy Complies
Enable sending ETH	equal	true	✓

1.7 Function

```
function updateState(  
    IKeeperRewards.HarvestParams calldata harvestParams  
)
```

Description

Updates a vault's state.

Parameter Sensitivity

LOW

- `harvestParams` - Parameters needed to update vault state, specifically used for keeper harvesting rewards. Typically retrieved from Stakewise subgraph or latest `rewardsUpdated` event from the relevant keeper contract.
 - `harvestParams.rewardsRoot` - The rewards merkle root.
 - `harvestParams.reward` - The Vault cumulative reward earned since the start. Can be negative in case of penalty/slashing.
 - `harvestParams.unlockedMevReward` - The Vault cumulative execution reward that can be withdrawn from shared MEV escrow. Only used by shared MEV Vaults.
 - `harvestParams.proof` - Proof to verify that Vault's reward is correct.

Access Control Recommendations

Parameter	Permission	Operator	Value	Policy Complies
<code>harvestParams</code>	Unscoped			✓

2. Lido Focused Aave V3 Instance

Removed from policy update as per request of ENS delegates

3. Balancer osETH/wETH StablePool Aura Deposit Vault

0x5F032f15B4e910252EDaDdB899f7201E89C8cD6b

3.1 Function

```
function getReward() external returns(bool)
```

Description

A staker calls this to get their allocated rewards.

Parameter Sensitivity

No Parameters

Access Control Recommendations

Unconstrained access to this function should pose no risk, as rewards are claimed for and sent to `msg.sender`.

3.2 Function

```
function getReward(  
    address _account,  
    bool _claimExtras  
) public updateReward(_account) returns(bool)
```

Description

Sends a staker their rewards, with the option of claiming extra rewards.

Parameter Sensitivity

LOW

- `_account` - Account to claim rewards for. Low sensitivity as the rewards for this account passed in are paid to the same account.

LOW

- `_claimExtras` - A boolean representing the desire to claim the extra 'child' rewards on top of the base rewards.

Access Control Recommendations

Parameter	Permission	Operator	Value	Policy Complies
<code>_account</code>	Unscoped			-
<code>_claimExtras</code>	Unscoped			✓

3.3 Function

```
function withdrawAndUnwrap(  
    uint256 amount,  
    bool claim  
) public returns(bool)
```

Description

Withdraws tokens from BaseRewardPool to `msg.sender`, with the option of claiming outstanding rewards too.

Parameter Sensitivity

LOW

- `amount` - Amount of tokens to withdraw.

LOW

- `claim` - A boolean representing the desire to claim the extra 'child' rewards on top of the base rewards.

Access Control Recommendations

Parameter	Permission	Operator	Value	Policy Complies
<code>amount</code>	Unscoped			✓
<code>claim</code>	Unscoped			✓

4. Balancer osETH/wETH-BPT Gauge Deposit

[0xc592c33e51A764B94DB0702D8BAf4035eD577aED](#)

Description

This smart contract implements a liquidity gauge authored by Curve Finance, which allows users to stake LP tokens, earn rewards in multiple tokens including BAL (Balancer's governance token), and participate in Balancer's governance system through voting power and boosting mechanisms.

4.1 Function

```
def withdraw(_value: uint256):
```

Description

This function allows users to withdraw LP tokens from the gauge.

Parameter Sensitivity

LOW

- `_value` - The amount of LP tokens to withdraw.

Access Control Recommendations

Parameter	Permission	Operator	Value	Policy Complies
<code>_value</code>	Unscoped			✓

4.2 Function

```
def deposit(_value: uint256):
```

Description

This function allows users to deposit LP tokens into the gauge.

Parameter Sensitivity

LOW

- `_value` - The amount of LP tokens to deposit into the gauge.

Access Control Recommendations

Parameter	Permission	Operator	Value	Policy Complies
<code>_value</code>	Unscoped			✓

4.3 Function

```
def claim_rewards()
```

Description

This function allows users to claim pending reward tokens from the gauge.

Parameter Sensitivity

No Parameters

Access Control Recommendations

Unconstrained access to this function should pose no risk, as rewards are claimed for and sent to `msg.sender`.

5. CowSwap Order Signer

Description

This contract serves as an aid to make CowSwap order signing possible for users of the Zodiac Roles Modifier.

5.1 Function

```
function unsignOrder(  
    GPv2Order.Data calldata order  
)  
  
library GPv2Order {  
    struct Data {  
        IERC20 sellToken;  
        IERC20 buyToken;  
        address receiver;  
        uint256 sellAmount;  
        uint256 buyAmount;  
        uint32 validTo;  
        bytes32 appData;  
        uint256 feeAmount;  
        bytes32 kind;  
        bool partiallyFillable;  
        bytes32 sellTokenBalance;  
        bytes32 buyTokenBalance;  
    }  
}
```

Description

This function unsigns a CowSwap order.

Parameter Sensitivity

HIGH

- `order.sellToken` - Token that is swapped from.
- `order.buyToken` - Token that is swapped to.
- `order.receiver` - Address that receives the proceeds of the swap. If this field the zero address then the user who signed the order is receiver.

MED

- `order.sellAmount` - Amount of sellToken that is sold.
- `order.buyAmount` - Amount of buyToken that is bought.

Both parameters above also carry information on the slippage of a trade. E.g \$100 of ETH (sellAmount) sold for at least 99 USDC (buyAmount) implies an allowed slippage of 1%. While independent solvers compete to fill a trade at the best price possible, there is still some risk in allowing a manager the ability to define the size of this slippage as we can't guarantee solvers won't try execute at higher slippage amounts if possible.

LOW

- `order.validTo` - UNIX timestamp (in seconds) until which the order is valid up until.
- `order.appData` - Extra information about the order. Not enforced by the smart contract outside of signature verification (may be used for referrals etc).
- `order.kind` - buy or sell order.
- `order.partiallyFillable` - Determine if the order is [partially fillable \(true\)](#) or [fill-or-kill \(false\)](#)
- `order.sellTokenBalance` - keccak256 hash of the balance location where the sellToken is withdrawn. Possible locations: `erc20` , `external` , `internal` .
- `order.buyTokenBalance` - keccak256 hash of the balance location where the buyToken is deposited. Possible locations: `erc20` , `external` , `internal` .
- `order.feeAmount` - Amount of fees paid in sellToken. If a manager tried to set an unreasonably high fee amount the protocol would [cap it at 1%](#) of the swap volume.

Value	Description
erc20	User's ERC-20 balance via approvals given to the GPv2VaultRelayer (default)
external	User's ERC-20 balance via approvals given to the Balancer vault
internal	User's internal Balancer vault balance

Access Control Recommendations

This recommendation is in line with the previously scoped `signOrder` function.

Parameter	Permission	Operator	Value	Policy Complies
order.sellToken	Scoped	One of	<i>sellTokens</i>	✓
order.buyToken	Scoped	One of	<i>buyTokens</i>	✓
order.receiver	Scoped	Equal	AVATAR	✓
order.sellAmount	Unscoped			✓
order.buyAmount	Unscoped			✓
order.validTo	Unscoped			✓
order.appData	Unscoped			✓
order.feeAmount	Unscoped			✓
order.kind	Unscoped			✓
order.partiallyFillable	Unscoped			✓
order.sellTokenBalance	Unscoped			✓
order.buyTokenBalance	Unscoped			✓

<i>sellTokens</i>	<i>buyTokens</i>
CRV	
DAI	DAI
USDT	USDT
BAL	
AURA	
RPL	
osETH	
CVX	
ETHx	
COMP	
rETH	rETH
SWISE	
wstETH	wstETH
LDO	

<i>sellTokens</i>	<i>buyTokens</i>
WETH	WETH
ankrETH	
USDC	USDC
stETH	stETH

Option	Operator	Value	Policy Complies
Enable Delegate Calls	equal	true	✓

The CowSwap order signer is not usable without enabling delegate call

Modifications Audit

1. Aave V3 Pool Contract

0x87870bca3f3fd6335c3f4ce8392d69350b4fa4e2

1.1 Function

```
function setUserUseReserveAsCollateral(
    address asset,
    bool useAsCollateral
)
```

Requested Modifications

Change	Parameter	Modification	Risk	Reasoning
ADDED	asset	Add osETH as collateral asset	LOW	Allows manager to set specific assets as collateral

1.2 Function

```
function supply(
    address asset,
    uint256 amount,
    address onBehalfOf,
    uint16 referralCode
) public virtual returns (uint256)
```

Requested Modifications

Change	Parameter	Modification	Risk	Reasoning
ADDED	asset	Add osETH as collateral asset	LOW	Restricts supply to specific assets, improving control over asset management

1.3 Function

```
function withdraw(
    address asset,
    uint256 amount,
    address to
) public virtual returns (uint256)
```

Requested Modifications

Change	Parameter	Modification	Risk	Reasoning
ADDED	asset	Add osETH as collateral asset	LOW	Restricts withdrawals to specific assets.

2. Aura Booster

0xA57b8d98dAE62B26Ec3bcC4a365338157060B234

Function

```
function deposit(uint256 _pid, uint256 _amount, bool _stake) public nonReentrant returns(bool)
```

Requested Modifications

Change	Parameter	Modification	Risk	Reasoning
ADDED	_pid	Allow pool ID 179	LOW	Allows manager to deposit osETH/WETH BPT to corresponding Aura pool.

3. Aura Reward Pool Deposit Wrapper

0xB188b1CB84Fb0bA13cb9ee1292769F903A9feC59

Function

```
function depositSingle(
    address _rewardPoolAddress,
    IERC20 _inputToken,
    uint256 _inputAmount,
    bytes32 _balancerPoolId,
    IVault.JoinPoolRequest memory _request
)
```

Requested Modifications

Change	Parameter	Modification	Risk	Reasoning
ADDED	_rewardPoolAddress	Must equal osETH/wETH StablePool Aura Deposit Vault	LOW	Restricts deposits to a specific reward pool
ADDED	_balancerPoolId	Must equal 0xdacf5fa19b1f720111609043ac67a9818262850c0000000000000000000000635	LOW	Limits deposits to

Change	Parameter	Modification	Risk	Reasoning
				the osETH/WETH balancer pool
ADDED	_inputToken	Add ability to deposit WETH	LOW	Restricts input tokens to WETH
ADDED	_inputToken	Add ability to deposit osETH	LOW	Restricts input tokens to osETH

4. Balancer Vault

0xba1222222228d8ba445958a75a0704d566bf2c8

4.1 Function

```
function exitPool(
    bytes32 poolId,
    address sender,
    address payable recipient,
    ExitPoolRequest memory request
)
```

Requested Modifications

Change	Parameter	Modification	Risk	Reasoning
ADDED	poolId	Add allowed pool ID: 0xdacf5fa19b1f720111609043ac67a9818262850c000000000000000000000635	LOW	Extend allow list of pools to 'exit' from with osETH/WETH

4.2 Function

```
function joinPool(
    bytes32 poolId,
    address sender,
    address recipient,
    JoinPoolRequest memory request
)
```

Requested Modifications

Change	Parameter	Modification	Risk	Reasoning
ADDED	poolId	Add allowed pool ID: 0xdacf5fa19b1f720111609043ac67a9818262850c000000000000000000000635	LOW	Extend allow list of pools to 'join' to osETH/WETH pool

4.3 Function

```
function swap(  
    SingleSwap memory singleSwap,  
    FundManagement memory funds,  
    uint256 limit,  
    uint256 deadline  
) external payable returns (uint256)
```

Requested Modifications

Change	Parameter	Modification	Risk	Reasoning
ADDED	singleSwap.poolId	Add poolId: 0xdacf5fa19b1f720111609043ac67a9818262850c0000000000000000000000635	LOW	Allow swaps on osETH/WETH pool
ADDED	singleSwap.assetIn	Add ability to sell WETH	LOW	Allow swapping from WETH tokens
ADDED	singleSwap.assetIn	Add ability to sell osETH	LOW	Allow swapping from osETH tokens
ADDED	singleSwap.assetOut	Add ability to buy WETH	LOW	Allow swapping to WETH tokens
ADDED	singleSwap.assetOut	Add ability to buy osETH	LOW	Allow swapping to osETH tokens

5. Balancer Minter Contract

0x239e55F427D44C3cc793f49bFB507ebe76638a2b

Function

```
function mint(address gauge)
```

Requested Modifications

Change	Parameter	Modification	Risk	Reasoning
ADDED	gauge	Add address osETH/WETH gauge	LOW	Allows manager to mint rewards for the osETH/WETH gauge

6. Cowswap Order Signer

0x23dA9AdE38E4477b23770DeD512fD37b12381FAB

Function

```
function signOrder(  
  GPv2Order.Data calldata order,  
  uint32 validDuration,  
  uint256 feeAmountBP  
)
```

Requested Modifications

Change	Parameter	Modification	Risk	Reasoning
ADDED	order.sellToken	Add osETH to sell list	LOW	Restricts sell tokens to approved list, reducing potential for unauthorized token sales
ADDED	order.sellToken	Add RPL to sell list	LOW	Restricts sell tokens to approved list, reducing potential for unauthorized token sales

7. Curve Deposit & Stake Zap

0x56c526b0159a258887e0d79ec3a80dfb940d0cd7

Function

```
def deposit_and_stake(  
  deposit: address,  
  lp_token: address,  
  gauge: address,  
  n_coins: uint256,  
  coins: DynArray[address, MAX_COINS],  
  amounts: DynArray[uint256, MAX_COINS],  
  min_mint_amount: uint256,  
  use_underlying: bool,  
  use_dynarray: bool,  
  pool: address,  
)
```

Requested Modifications

Change	Parameter	Modification	Risk	Reasoning
REPLACE	coins	Changed allowed coins array length from 5 to 2	LOW	Previous scoping was incorrectly adding zero addresses to the parameter.

8. Uniswap V3 Router 2

0x68b3465833fb72a70ecdf485e0e4c7bd8665fc45

Function

```
function exactInputSingle(ExactInputSingleParams memory params) external payable returns (uint256 amountOut)
```

Requested Modifications

Change	Parameter	Modification	Risk	Reasoning
ADDED	params.tokenIn	Add osETH to sell list	LOW	Restricts sell tokens to approved list, reducing potential for unauthorized token sales
ADDED	params.tokenIn	Add RPL to sell list	LOW	Restricts sell tokens to approved list, reducing potential for unauthorized token sales

Address Verification & Approval Audit

1. Context

This report provides an analysis of the new contract addresses referenced in the New Functions Audit sub-heading. The purpose is to verify that these addresses supplied are indeed the intended addresses for permission granting activities and are non-malicious. Our analysis covers both token addresses and protocol addresses, examining them across several key areas.

This verification will defend against both human error and address poisoning attacks and token/protocol imitation scams. On top of this, this section of the report deals with new ERC20 approvals contained in the policy update.

2. Definitions

Analysis Type Definitions

Analysis Type	Description
Swap Analysis	This analysis checks if the token can be swapped on decentralized exchanges. It ensures that the token is not a honeypot which can be bought but not sold
Contract Analysis	Checks ownership renounced or source does not contain an owner contract. Checks contract creator not authorised for special permissions and contract is verified
Holder Analysis	This analysis examines the distribution of top token holders. It looks for concentration of tokens among a few addresses, which could indicate a risk of market manipulation or centralisation.
Liquidity Analysis	This analysis checks the liquidity available for the token on-chain. This aids detecting scam tokens which often have weak liquidity.
Immutable	This analysis checks for proxy contracts. If a token is upgradeable the risk of interacting with it changes with each implementation upgrade
External Confirmations	This analysis involves cross-referencing the token or protocol address with external sources such as official documentation, protocol front end address reference and trusted third-party services like blockexplorers and

Analysis Type	Description
	analytics providers.

3. New Token Addresses

3.1 osETH Token Contract

0xf1c9acdc66974dfb6decb12aa385b9cd01190e38

Analysis Type	Passed	Comments
Swap Analysis	✓	
Contract Analysis	✓	
Holder Analysis	✓	
Liquidity Analysis	✓	
Immutable	✓	
External Confirmations	✓	

3.2 RPL Token Contract

0xd33526068d116ce69f19a9ee46f0bd304f21a51f

Analysis Type	Passed	Comments
Swap Analysis	✓	
Contract Analysis	✓	
Holder Analysis	✓	
Liquidity Analysis	✓	
Immutable	✓	
External Confirmations	✓	

4. New Protocol Addresses

4.1 Lido Focused Aave v3 Instance

Removed from policy update as per request of ENS delegates

4.2 Stakewise v3 ETH Genesis Vault

0xAC0F906E433d58FA868F936E8A43230473652885

Analysis Type	Passed	Comments
External Confirmations	✓	

4.3 Balancer Pool Token WETH/osETH

0xDACf5Fa19b1f720111609043ac67A9818262850c

Analysis Type	Passed	Comments
External Confirmations	✓	


4.4 osETH/WETH-BPT Gauge

[0xc592c33e51A764B94DB0702D8BAf4035eD577aED](#)

Analysis Type	Passed	Comments
External Confirmations	✓	

4.5 Balancer osETH/wETH StablePool Aura Deposit Vault

[0x5F032f15B4e910252EDaDdB899f7201E89C8cD6b](#)

Analysis Type	Passed	Comments
External Confirmations		No etherscan label for contract/deployer, no reference in official Aura docs but is explicitly referenced on Aura frontend.

5. Approvals

Contract	Target	Verified
RPL	Cow GpV2VaultRelayer	✓
RPL	Uniswap v3: Router 2	✓
BPT: osETH/WETH	Balancer osETH/wETH-BPT Gauge	✓
BPT: osETH/WETH	Aura Booster	✓
osETH	Cow GpV2VaultRelayer	✓
osETH	Aura Reward Pool Deposit Wrapper	✓
osETH	Uniswap v3: Router 2	✓
osETH	Balancer Vault	✓
osETH	Aave Pool v3	✓

Please note, only new addresses added to the policy are verified within this report. Addresses previously referenced in the existing policy will be assumed to have been verified already.

6. Glossary

- **Avatar:** In the context of Gnosis Safe and Zodiac, the Avatar is the account that ultimately owns assets and executes transactions. It's typically a Gnosis Safe multi-signature wallet.
- **Zodiac Roles Modifier:** A Zodiac module that allows for fine-grained access control on a Gnosis Safe.
- **Scoped:** When calling a function, the caller can only pass scoped values to the specific parameter.
- **Unscoped:** When calling a function, the caller has the discretion to pass **ANY** value to a specific parameter.
- **aToken:** Aave interest-bearing tokens that are minted and burned upon supply and withdraw of assets to/from the Aave protocol.
- **LP Token:** Liquidity Provider Token, representing a share in a liquidity pool.
- **BPT:** Balancer Pool Token, representing a share in a Balancer liquidity pool.
- **ERC20:** A standard interface for fungible tokens on the Ethereum blockchain.
- **Flash Loan:** A type of uncollateralized loan in DeFi where borrowed funds must be returned within the same transaction.

- **GPv2VaultRelayer:** A component of the CowSwap protocol.
- **MEV:** Maximal Extractable Value, the maximum value that can be extracted from block production in excess of standard block rewards and gas fees.
- **Slippage:** The difference between the expected price of a trade and the price at which the trade is executed.
- **Staking:** The process of locking up tokens, generally in return for rewards.
- **Address Poisoning:** A type of attack where malicious actors create addresses that closely resemble legitimate ones to trick users into sending funds to the wrong address.
- **Honeypot:** A scam where a token appears legitimate but is designed to prevent selling, trapping investors' funds.
- **Immutable:** Refers to a smart contract that cannot be altered once deployed.
- **Proxy Contract:** A smart contract that delegates calls to another contract, allowing for upgradeable contracts.
- **Gauge:** In DeFi, often refers to a contract that distributes rewards to users who provide liquidity or stake tokens.
- **Vault:** In DeFi, a smart contract that holds and manages user funds, often used for yield farming strategies.
- **Router:** In DeFi, a smart contract that determines the best path for a token swap across various liquidity pools.

7. Disclaimer

This audit report is provided for informational purposes only and should not be considered as financial, legal, or investment advice. The primary purpose of this report is to prevent potential misappropriation by the manager and ensure that DeFi actions are bonafide. It is not intended to serve as an official smart contract audit of the specific protocols utilized.

Key assumptions in this report include:

- Address verifications and existing policy scoping for the modified functions have already been verified in previous audits.
- The "Roles by Gnosis Guild" user interface, from which policy permissions were retrieved, is assumed to be a source of independent truth.

While every effort has been made to ensure the accuracy and thoroughness of this report, no guarantee is made as to its completeness or correctness. The information contained herein is based on the current state of knowledge and understanding as of the date of this report, and may become outdated or inaccurate due to subsequent developments in blockchain technology, smart contract ecosystems, or regulatory environments.

The auditors and ThirdGuard do not accept any liability for any direct, indirect, consequential, or other losses or damages arising out of or in connection with the use of this report or any information contained herein. Users of this report assume all risks associated with the use of any information provided.

This report does not constitute an endorsement of any project, token, or smart contract. Readers are strongly advised to conduct their own research and due diligence before making any decisions based on this report. The ultimate responsibility for any actions taken based on this information lies solely with the reader.

By using this report, you acknowledge that you have read, understood, and agree to the terms of this disclaimer.