

# SMART CONTRACT SECURITY AUDIT REPORT

For Rollup.Finance

24 April 2023



## Table of Contents

1. Overview.....	4
2. Background .....	5
2.1 Project Description .....	5
2.2 Audit Range.....	6
3. Project contract details.....	10
3.1 Contract Overview .....	10
3.2 Contract details .....	19
4. Audit details .....	76
4.1 Findings Summary .....	76
4.2 Risk distribution .....	77
4.3 Risk audit details .....	79
4.3.1 Administrator permissions.....	79
4.3.2 Same address judgment .....	80
4.3.3 Logic Design Flaw .....	82
4.3.4 Redundant codes.....	85
4.3.5 Reentry Attack.....	89
4.3.6 Variables are updated .....	90
4.3.7 Floating Point and Numeric Precision.....	90
4.3.8 Default Visibility .....	91
4.3.9 tx.origin authentication .....	91
4.3.10 Faulty constructor.....	92
4.3.11 Unverified return value .....	92
4.3.12 Insecure random numbers.....	93
4.3.13 Timestamp Dependency.....	93
4.3.14 Transaction order dependency .....	94
4.3.15 Delegatecall.....	94
4.3.16 Call.....	95
4.3.17 Denial of Service .....	95
4.3.18 Fake recharge vulnerability .....	96

---

4.3.19 Short Address Attack Vulnerability .....	96
4.3.20 Uninitialized storage pointer.....	97
4.3.21 Frozen Account bypass .....	97
4.3.22 Uninitialized .....	97
4.3.23 Integer Overflow.....	98
5. Security Audit Tool.....	99

---

## 1. Overview

On Apr 15, 2023, the security team of Lunaray Technology received the security audit request of the **ROLLUP.FINANCE project**. The team completed the audit of the **ROLLUP.FINANCE smart contract** on Apr 24, 2023. During the audit process, the security audit experts of Lunaray Technology and the ROLLUP.FINANCE project interface Personnel communicate and maintain symmetry of information, conduct security audits under controllable operational risks, and avoid risks to project generation and operations during the testing process.

Through communication and feedback with ROLLUP.FINANCE project party, it is confirmed that the loopholes and risks found in the audit process have been repaired or within the acceptable range. The result of this ROLLUP.FINANCE smart contract security audit:

**Passed**

Audit Report Hash:

6E89109793737F48A407EACE0218E572A2D4D06EB61AE304B57B1C65932B502F

---

## 2. Background

### 2.1 Project Description

---

<b>Project name</b>	Rollup.Finance
<b>Contract type</b>	Spot and perpetual social trading
<b>Code language</b>	Solidity
<b>Public chain</b>	zkSync
<b>Project website</b>	<a href="https://rollup.finance">https://rollup.finance</a>
<b>Contract file</b>	YieldToken.sol,USDR.sol,LP.sol,WETH.sol,BaseToken.sol,MintableBaseToken.sol,FaucetToken.sol,Multicall.sol,TokenManager.sol,Timelock.sol,Governable.sol,Reader.sol,VaultReader.sol,BalanceUpdater.sol,BatchSender.sol,RewardReader.sol,OrderBookReader.sol,DexV3Aggregator.sol,FastPriceEvents.sol,CustomV3Aggregator.sol,ConstantV3Aggregator.sol,FastPriceFeed.sol,VaultWrapper.sol,PositionRouter.sol,VaultPriceFeed.sol,PositionManager.sol,ShortsTracker.sol,OrderBook.sol,Vault.sol,Router.sol,BasePositionManager.sol,RewardTracker.sol,RewardRouterV1.sol,RewardDistributor.sol,ReferralStorage.sol,ReferralReader.sol

---

**Brief introduction** Rollup.Finance is a decentralized perpetual contract protocol based on zkSync. It offers trading in multiple derivative contracts, promises high returns and provides a liquidity solution for pledged notes . It aims to create the largest multi-decentralized derivatives trading platform, supporting multiple currencies, supporting zero slippage, and addressing capital utilization efficiency and liquidity issues.

---

## 2.2 Audit Range

### Smart contract file name and corresponding SHA256:

Name	SHA256
YieldToken.sol	934EB8FEE29BD2718D1BB3AEA0A1C2385216AF2CA298068 422106172044C0B31
USDR.sol	8AF1706CC15BBE5A91CC5E79014AFA265F32DA97C973F96 558E9E65AA96A7BBF
LP.sol	EBEA9C95469DE9A86F58A1BDB0FC0D1CB9CC248520AC38 7036D86304B19BB551
WETH.sol	05F87C74ECFE266BC3D70534ADA651DDBA2BE1CEE82031 A36B11F1A51E0D3755
BaseToken.sol	8B8BD15090EB891BB95AC0DCCA9F95EF2465E010608202D 5A1D7B99D2AF24AA8
MintableBaseToken.sol	5DC1C857DEBA7C4250011273C58E9967EBBA2512E17F0E5 FEB1D3A85E0DB240D
FaucetToken.sol	DC0CC81B20EC0FDC012248D39C7041A27FD327D463FA00 72B7411E4FF7B0DA4C
Multicall.sol	BE0A4CBE03A9C47D464E28405A772BD702EFF80E6ED97A0 A938C733DAADDDBB57
TokenManager.sol	B3F53C9F973AC600D5A4CD877230FF69179F094DDA1A32C 12202612F6620D2E1
Governable.sol	A002AFCEF81A5743C542E2BBF1E750A311BF87F32F8DD1A 8E0CA8F3E346012AB

---

Reader.sol	76AABF02BD8C349CC13F7F3D0958E2F0EB7289DB451C53F68F57CA6E62F036E6
VaultReader.sol	ADDA6E32BAE2CB44ACD5A8AEE1CCF68C405FFA3C7373C90B7296481792871FC8
BalanceUpdater.sol	759139F23E3F3424076E5AF2FFF9B0581906E97297AB72443AB82332617D1965
BatchSender.sol	3E19A8036C2496BFC1AA4A0F939A43B6FF0B862B0AFB3D73509F89C996165C6F
RewardReader.sol	C87F4E8CFB4CEEE5CA0EE8BBAA46A198AFA0D0841AC3CB9A9054A1F05A16011
OrderBookReader.sol	CC39BE8F62078DB529F4D7D658FDCED2994B7146D83F8FE965640E565F6EBF2D
DexV3Aggregator.sol	Aefd194DD07340BA85AB1F776D90B1B39BF951F39946582E5AC4506CE894A400
FastPriceEvents.sol	6BD8D2795D3C9191CB4ACFBD0CA15B612EA793562CB1286A195C47CD08F1BE25
CustomV3Aggregator.sol	8EB250A2AC820D75E7CAE934954FE7D3C18D8C2DD8E1D093EB56D7C674257A4E
ConstantV3Aggregator.sol	EC610AA475FFCFFD21BF657819DD3031D41F98F1201B7B63E20FDF1093A40134
FastPriceFeed.sol	108F50885C9893BB2F7450A80F2A52285B9397B57AE1BA645EF80356A1B1366C
LpManager.sol	8E2E3A74FE00D34DEE87A9877D6F676AA3777D7B7F785E818E6230F2B5C4A18F

---

---

VaultWrapper.sol	3616E780E153CD79BA2623A3D2BD09834D6F790A8B0713 84EFDB9724997C05B3
PositionRouter.sol	ED4A4EC4164799BA8F637080B70A1C4B47B07E6683F8C34 637110A100E06B994
VaultPriceFeed.sol	A9FCFDACBD34023C562B0852DDCA7FE103926145F8C3A2 028760F8173BF4958F
PositionManager.sol	C4C97BCCFF56693DB783215C49D7BB06A0F289E9E6AEBFD DAD2C10F1CAAB9073
ShortsTracker.sol	8D1985A308CE98951F4BD7E11DFB119EE38DEF86B2C2BC3 31CB0F518A1B23FA6
OrderBook.sol	05A2B83DEAF0C37F66416B3EC28F0D84ACD551808D10FC1 9F007E5CA80815937
Vault.sol	01601536D9D03F7E5E7142747D3A73EEABB319FDEEC9597 CAC336DC7733C04BC
Router.sol	C0165FAD09F57065B35426D548212B057534F08CF68F540 62973B28D264FD15B
BasePositionManager.sol	9700AFDC5F423F0B5CF84788F7A71B3F60D954EC8FAA7F6 6002746B7A50B0E67
RewardTracker.sol	27382DB51E3C7232C9EEE3D688B818B75F9F1405DBB9926 64A80DC379AE0E336
RewardRouterV1.sol	179F52121E97ED4C5B0200F69CA8CB06E2DE1E1FC97EE28 53D0A40D0C4D99C10
RewardDistributor.sol	4CF95FE5BABD593BB73AC74EF5B325F2C3BF3A7C89F639F 03AEC9137BED8EE4A

---



---

ReferralStorage.sol

79A5EB739D4E0DDCD5B68CCE7DF116E1B184C429D83E55  
1E9A2C58ABE1C0C547

---

ReferralReader.sol

50E14ACA0B5FF2A885F526B5121A72CC3A5C9231BF31E0D  
64452C56E04079DD2

---

## 3. Project contract details

### 3.1 Contract Overview

#### Valut Contract

The Valut contract is the base contract for the whole system and is mainly used by other contracts. The roles that call the contract are Gov, Manager, liquidator and normal user. The functions that can be called by Gov are initialization function, set external interface contract address, set management mode, set clearer whitelist, set contract exchange switch, set contract leverage switch, update whitelist token address and quantity, set interest rate, set Token configuration, etc.; the contracts that can be called by Manager are buying and selling USDR, calculating reserve, adding and subtracting positions, calculating rewards, etc. Users can call functions such as setting Router, token exchange, querying token information, querying position information, getting interest rate, calculating fees for buying and selling USDR or exchange, querying liquidation, etc. The authority to call liquidation functions is set by Gov, when Gov sets the liquidation status to private mode, only the liquidator can perform. If the liquidation fee does not exceed the collateral, part of the collateral will be liquidated, and the remaining collateral will be set as the upper limit of the collateral.

#### BasePositionManager Contract

This smart contract is a basic position management contract, mainly used to manage the basic positions of the contract (including long positions and short positions), as well as position increase, decrease and other related operations, all functions of the contract are visible internally, the user can not directly call, but indirectly through the PositionManager contract to call.

#### LpManager Contract

The main function of the contract is to manage liquidity; the contract implements functions such as adding liquidity, and removing liquidity. It also contains some variables that can be modified by the administrator and public methods that can access some of the data. There are two types of liquidity addition, native chain token and other token, and two types of liquidity removal. The Handler list of users, set by the administrator, can add and remove liquidity from any other user.

## **OrderBook Contract**

OrderBook contract implements an order management that can be used for matching and processing of trades. It implements the functions of creating, acquiring, updating, and executing three different types of orders: trading, position addition, and position reduction. It also contains some functions such as setting contract parameters by the administrator.

## **PositionManager Contract**

This contract is mainly used for position management and inherits the BasePositionManager contract, which mainly implements the functions of adding and subtracting positions that require the Partners authority, and supports ETH as input coins and output tokens; secondly, it implements the functions of executing exchange, adding and subtracting orders for the OrderKeeper authority, and the functions of liquidating positions that require the Liquidator authority is required for liquidation.

## **PositionRouter Contract**

The main function of this contract is to execute position addition and reduction operations. The contract inherits the BasePositionManager contract, and the contracts called are BasePositionManager, Valut contract, etc. The main functions of the contract are Admin to set position manager, minimum execution fee, leverage status, delay value, etc. The main functions of the position manager are to execute bulk position addition and reduction operations; the functions of ordinary users are to create and cancel position addition and reduction.

## **Router Contract**

The main function of this contract is for the user to perform the exchange operation by adding and subtracting positions through this contract, mainly by calling the Valut contract to perform the operation. The main functions of the contract are user authorization, transfer via Router, transfer Token to the pool, token exchange, add/drop positions directly via the specified Token, add/drop positions via ETH, reduce positions and exchange them to the user specified Token or ETH, etc.

### **ShortsTracke Contract**

The main function of this contract is to track and calculate the actual price of the shorted tokens and the user's profit calculation, this contract inherits the Governable contract. The main functions of the contract are to set Handler administrator, set token initialization data; administrator update global data; user can query actual profit and loss, subsequent average price and position, etc.

### **VaultPriceFeed Contract**

The contract mainly provides a price feeding mechanism, which is called by other contracts to query the Token price. The main functions of the contract are Gov set price update time, chain flag, AMM status, Token address, Token configuration, etc.; users can query Token price, get initial price, get on-chain price, get recent price, get secondary price, etc.

### **FastPriceFeed Contract**

The contract provides a second layer of price sources, fast price updates, and permission control. In practice, the contract provides an external function `getPrice` for external contract calls to provide price data, implements various functions to set token prices, is limited to contracts with Updater permissions, and includes some administrator-modifiable parameters.

### **FastPriceEvents Contract**

The main logic of the contract is used to keep track of price sources and trigger `PriceUpdate` events when the price sources are updated. By using this contract, it is possible to record the update history of price sources on the chain and ensure that only verified price sources can make calls to the contract.

### **FastPriceEvents Contract**

The contract mainly implements the off-chain price data to be verified and stored by an on-chain smart contract, providing an on-chain price prognosticator that can update the price data, which can be called by an on-chain contract to get the current price of a certain asset.

### **BalanceUpdater Contract**

The main logic is to update the balance of the specified token `_token` in the specified Vault contract and transfer a specific number of USDR tokens to the Vault contract, then sell them and send the proceeds back to the caller's address.

### **BatchSender Contract**

The contract is mainly designed to implement the batch transfer function, i.e., by calling the `send()` or `sendAndEmit()` functions, you can send `_token` tokens to multiple addresses in the `_accounts` array and specify the number received by each address. Among other things, the `sendAndEmit()` function can specify the `_typeId` parameter that identifies the type of this batch transfer in the BatchSend event. In addition, the contract has some administrative features, such as only the addresses added to the `isHandler` mapping have permission to call the transfer function. Also, the `setHandler()` function can only be called by the administrator of the contract (i.e., the `onlyGov` modifier inherited through the Governable contract) to control the permissions of the address.

### **Reader Contract**

The contract provides `getMaxAmountIn`, `getAmountOut` and `getFeeBasisPoints` functions to query the maximum number of Tokens to be exchanged, the expected number of Tokens to be exchanged and the fee percentage.

## RewardReader Contract

The contract implements a number of functions for reading information, including getting the maximum amount of a token that can be exchanged into a Vault, getting the transaction fee percentage, getting the yield, getting the token balance, and so on. The beginning of the contract defines some constants, such as the denominator of base points, price precision, etc. Also, the contract inherits from the Governable contract, so it has administrator privileges and only the administrator can call the setConfig function to set the hasMaxGlobalShortSizes value.

## VaultReader Contract

The contract contains two functions getVaultTokenInfoV3 and getVaultTokenInfoV4. Both of these functions are read-only. These functions read and return information about the Vault contract and other contracts. Initially, it is assumed that the purpose of this contract is to allow the user to query information about the tokens in the Vault contract. This information includes: the balance of each token in the Vault contract, the number of reserved, the current USDr number, the number that can be redeemed, the weight of the tokens, the number of buffers, the maximum USDr number, the global short size, the minimum and maximum price of the tokens, the guaranteed USD quantity, and token price information.

## ReferralStorage Contract

The contract implements storing and managing the relationship between the referrer and the referee. The basic logic is to define a structure containing rebate and discount percentages that can be used to specify the rebate and discount percentages for different referrers. The contract provides a number of methods, such as setting specific rebate and discount percentages for the referrer, setting referral codes for the referee, and so on. In addition, the contract provides detailed events for recording the details of the above mentioned operations.

### ReferralReader Contract

This contract mainly contains a public view function called `getCodeOwners`. This function retrieves the addresses of the associated invitation code owners from the `ReferralStorage` contract and stores these addresses in the returned array.

### RewardDistributor Contract

The contract implements a reward distribution mechanism that allows a specific contract (i.e., a reward tracker) to withdraw a certain number of tokens as rewards from that contract at a specified time interval. The contract manages a pool of `Token` rewards and can set the time interval for rewards and the number of rewards in each time interval. It also keeps track of when the last reward was assigned and can calculate the number of rewards that should have been assigned since the last reward assignment. When the reward tracker calls the `distribute()` function, the corresponding tokens will be extracted from the reward pool based on the calculated number of rewards and transferred to the address of the reward tracker. In addition, the administrator of the contract can change the time interval and the number of rewards in each interval, as well as update the last reward distribution time. In addition to this, the contract provides a `withdrawToken` function, which is called by the `Gov` permission only, to help the user extract the wrongly sent tokens from the contract.

### RewardRouterV1 Contract

The contract is a routing contract that allows users to add LP liquidity to the `Liquidity Provider (LP)` pool and remove LP liquidity, both supporting chain-native tokens and other `ERC20` `Token`. It also implements the ability for users to withdraw their earnings, supporting withdrawals as chain-native tokens. The contract also includes the option to withdraw tokens to the user's account in case they accidentally send them to this contract.

### **Governable Contract**

Governable contract is a project management contract with the function of setting privileged role gov.

### **TokenManager Contract**

The TokenManager contract is the management contract of this project, which executes the multi-signature authorization and execution operations in the contract. The main logic is that the requester initiates a signature request, the administrator signs it, and then executes the operation when the minimum number of signatures is met. The main operations performed are token authorization, NFT authorization, NFT transfer, setting gov and so on.

### **BaseToken Contract**

BaseToken contract is an ERC-20 token contract. This contract mainly realizes the functions of token minting and destruction and transfer. Administrators can add and delete accounts without pledging and help designated accounts to receive rewards; users can check the number of pledges and transfer tokens, etc.

### **FaucetToken Contract**

According to the contract logic, this contract is a faucet token, and users can receive the relevant tokens for free.

### **LP Contract**

The LP contract inherits from the MintableBaseToken contract. The main function of this contract is to query and return the BaseToken contract token symbols.



### **MintableBaseToken Contract**

The MintableBaseToken contract inherits the BaseToken contract, and the main function is for the administrator to set up the minters, who can perform token minting and destruction operations.

### **USDR Contract**

The USDR contract inherits the YieldToken contract, and the main function is for the administrator to set the vault address, which can perform token minting and destruction operations.

### **WETH Contract**

The WETH contract is an ERC-20 token contract that allows users to access funds and transfer them.

### **YieldToken Contract**

The contract implements the IERC20 and IYieldToken interfaces and provides some special features for specific types of ERC20 tokens. It provides the ability to add or remove administrator addresses, and the administrator has some special privileges. In addition, it supports the ability to withdraw other ERC20 tokens, control which accounts' tokens can be pledged or redeemed, and withdraw proceeds from participating YieldTracker. Finally, it implements the standard ERC20 transfer and authorization transfer methods, and updates the account's rewards before transfer.

### **VaultWrapper Contract**

The contract provides the following functions: set leverage, set fees, enable/disable leverage. The contract sets fees by calling the setFees function of the Vault contract, and can enable/disable leverage as needed. In addition, the contract can also set whether the isLeverageEnabled flag should be toggled to control whether leverage is enabled or not. The specific function of the contract, as inferred from the contract logic, is to provide flexible fee and leverage control options for the Vault contract.

### **ConstantV3Aggregator Contract**

This contract implements the `AggregatorV2V3Interface` interface. Its main role is to provide a fixed price predictor, store the price via the `latestAnswer` variable, and return that price information via the interface function.

### **DexV3Aggregator Contract**

The contract is an aggregator, which takes a weighted average of the prices from multiple price sources to arrive at the price of the token. The contract uses the `Governable` module to implement permission control. The constructor is passed in the address of the token and the number of decimal places of the token. The contract has the ability to add and remove price sources. Each token can have multiple price sources, each with an array of weights and paths. When calculating the price, all the price sources are iterated, the price is obtained using the `IQuotePrice` interface, and the final price is calculated as a weighted average of the weights. The contract implements the `AggregatorV2V3Interface` interface of Chainlink, which supports querying the latest price, timestamp, round data, etc.

### **DexV3AggregatorV2 Contract**

The function of the contract is to provide a price aggregation service for a specific token, which can fetch prices from multiple price sources and calculate a weighted average price for that token. The contract's administrator can add or remove price sources, but each price source must be authorized by the token contract.

### 3.2 Contract details

#### Multicall Contract

Name	Parameter	Attributes
aggregate	Call[] memory calls	public
getEthBalance	address addr	public
getBlockHash	uint256 blockNumber	public
getLastBlockHash	none	public
getCurrentBlockTimestamp	none	public
getCurrentBlockDifficulty	none	public
getCurrentBlockGasLimit	none	public
getCurrentBlockCoinbase	none	public

#### YieldToken Contract

Name	Parameter	Attributes
setGov	address _gov	onlyGov
setInfo	string _name string _symbol	onlyGov
setYieldTrackers	address[] memory _yieldTrackers	onlyGov
addAdmin	address _account	onlyGov
removeAdmin	address _account	onlyGov
withdrawToken	address _token address _account uint256 _amount	onlyGov
setInWhitelistMode	bool _inWhitelistMode	onlyGov
setWhitelistedHandler	address _handler bool _isWhitelisted	onlyGov
addNonStakingAccount	address _account	onlyAdmin
removeNonStakingAccount	address _account	onlyAdmin

recoverClaim	address_account address_receiver	onlyAdmin
claim	address_receiver	external
totalStaked	none	external
balanceOf	address_account	external
stakedBalance	address_account	external
transfer	address_recipient uint256_amount	external
allowance	address_owner address_spender	external
approve	address_spender uint256_amount	external
transferFrom	address_sender address_recipient uint256_amount	external
_mint	address_account uint256_amount	internal
_burn	address_account uint256_amount	internal
_transfer	address_sender address_recipient uint256_amount	private
_approve	address_owner address_spender uint256_amount	private
_updateRewards	address_account	private

## USDR Contract

Name	Parameter	Attributes
addVault	address_vault	onlyGov
removeVault	address_vault	onlyGov
mint	address_account uint256_amount	onlyVault
Burn	address_account uint256_amount	onlyVault

## LP Contract

Name	Parameter	Attributes
id	none	external

## WETH Contract

Name	Parameter	Attributes
deposit	none	public
withdraw	uint256 amount	public
name	none	public
symbol	none	public
decimals	none	public
totalSupply	none	public
balanceOf	address account	public
transfer	address recipient uint256 amount	public
allowance	address owner address spender	public

approve	address spender uint256 amount	public
transferFrom	address sender address recipient uint256 amount	public
increaseAllowance	address spender uint256 addedValue	public
decreaseAllowance	address spender uint256 subtractedValue	public
_transfer	address sender address recipient uint256 amount	internal
_mint	address account uint256 amount	internal
_burn	address account uint256 amount	internal
_approve	address owner address spender uint256 amount	internal
_beforeTokenTransfer	address from address to uint256 amount	internal
_msgSender	none	internal

### BaseToken Contract

Name	Parameter	Attributes
setGov	address _gov	onlyGov
setInfo	string _name string _symbol	onlyGov
setYieldTrackers	address[] memory _yieldTrackers	onlyGov
addAdmin	address _account	onlyGov
removeAdmin	address _account	onlyGov
withdrawToken	address _token	onlyGov

	address _accoun uint256 _amount	
setInPrivateTransferMode	bool _inPrivateTransferMode	onlyGov
setHandler	address _handler bool _isActive	onlyGov
addNonStakingAccount	address _account	onlyAdmin
removeNonStakingAccount	address _account	onlyAdmin
recoverClaim	address _account address _receiver	onlyAdmin
claim	address _receiver	external
totalStaked	none	external
balanceOf	address _account	external
stakedBalance	address _account	external
transfer	address _recipient uint256 _amount	external
allowance	address _owner address _spender	external
approve	address _spender uint256 _amount	external
transferFrom	address _sender address _recipient uint256 _amount	external
_mint	address _account uint256 _amount	internal
_burn	address _account uint256 _amount	internal
_transfer	address _sender address _recipient uint256 _amount	private
_approve	address _owner address _spender uint256 _amount	private
_updateRewards	address _account	private

### MintableBaseToken Contract

Name	Parameter	Attributes
setMinter	address_minter bool_isActive	onlyGov
mint	address_account uint256_amount	onlyMinter
burn	address_account uint256_amount	onlyMinter

### FaucetToken Contract

Name	Parameter	Attributes
mint	address account uint256 amount	public
enableFaucet	none	public
disableFaucet	none	public
setDropletAmount	uint256 dropletAmount	public
claimDroplet	none	public
name	none	public
symbol	none	public
decimals	none	public
totalSupply	none	public
balanceOf	address account	public
transfer	address recipient uint256 amount	public
allowance	address owner address spender	public
approve	address spender uint256 amount	public
transferFrom	address sender	public



	address recipient uint256 amount	
increaseAllowance	address spender uint256 addedValue	public
decreaseAllowance	address spender uint256 subtractedValue	public
_transfer	address sender address recipient uint256 amount	internal
_mint	address account uint256 amount	internal
_burn	address account uint256 amount	internal
_approve	address owner address spender uint256 amount	internal
_beforeTokenTransfer	address from address to uint256 amount	internal
_msgSender	none	internal

### TokenManager Contract

Name	Parameter	Attributes
initialize	address[] memory _signers	onlyAdmin
signersLength	none	public
signalApprove	address _token address _spender uint256 _amount	onlyAdmin
signApprove	address _token address _spender uint256 _amount uint256 _nonce	onlySigner

approve	address_token address_spender uint256_amount uint256_nonce	onlyAdmin
signalApproveNFT	address_token address_spender uint256_tokenId	onlyAdmin
signApproveNFT	address_token address_spender uint256_tokenId uint256_nonce	onlySigner
approveNFT	address_token address_spender uint256_tokenId uint256_nonce	onlyAdmin
signalApproveNFTs	address_token address_spender uint256[] memory_tokenIds	onlyAdmin
signApproveNFTs	address_token address_spender uint256[] memory_tokenIds uint256_nonce	onlySigner
approveNFTs	address_token address_spender uint256[] memory_tokenIds uint256_nonce	onlyAdmin
receiveNFTs	address_token address_sender uint256[] memory_tokenIds	onlyAdmin
signalSetAdmin	address_target address_admin	onlySigner
signSetAdmin	address_target address_admin uint256_nonce	onlySigner
setAdmin	address_target address_admin	onlySigner

	uint256 _nonce	
signalSetGov	address _timelock address _target address _gov	onlyAdmin
signSetGov	address _timelock address _target address _gov uint256 _nonce	onlySigner
setGov	address _timelock address _target address _gov uint256 _nonce	onlyAdmin
_setPendingAction	bytes32 _action uint256 _nonce	private
_validateAction	bytes32 _action	private
_validateAuthorization	bytes32 _action	private
_clearAction	bytes32 _action uint256 _nonce	private

### Timelock Contract

Name	Parameter	Attributes
setAdmin	address _admin	onlyTokenMa nager
setExternalAdmin	address _target address _admin	onlyAdmin
setContractHandler	address _handler bool _isActive	onlyAdmin
initLpManager	none	onlyAdmin
initRewardRouter	address _rewardRouter	onlyAdmin
setKeeper	address _keeper bool _isActive	onlyAdmin
setBuffer	uint256 _buffer	onlyAdmin
setMaxLeverage	address _vault	onlyAdmin

	uint256_maxLeverage	
setFundingRate	address_vault uint256_fundingInterval uint256_fundingRateFactor uint256_stableFundingRateFactor	onlyKeeperAndAbove
setShouldToggleIsLeverageEnabled	bool_shouldToggleIsLeverageEnabled	onlyHandlerAndAbove
setMarginFeeBasisPoints	uint256_marginFeeBasisPoints uint256_maxMarginFeeBasisPoints	onlyHandlerAndAbove
setSwapFees	address_vault uint256_taxBasisPoints uint256_stableTaxBasisPoints uint256_mintBurnFeeBasisPoints uint256_swapFeeBasisPoints uint256_stableSwapFeeBasisPoints	onlyKeeperAndAbove
setFees	address_vault uint256_taxBasisPoints uint256_stableTaxBasisPoints uint256_mintBurnFeeBasisPoints uint256_swapFeeBasisPoints uint256_stableSwapFeeBasisPoints uint256_marginFeeBasisPoints uint256_liquidationFeeUsd uint256_minProfitTime bool_hasDynamicFees	onlyKeeperAndAbove
enableLeverage	address_vault	onlyHandlerAndAbove
disableLeverage	address_vault	onlyHandlerAndAbove
setIsLeverageEnabled	address_vault bool_isLeverageEnabled	onlyHandlerAndAbove
setTokenConfig	address_vault address_token uint256_tokenWeight uint256_minProfitBps uint256_maxUsdrAmount uint256_bufferAmount	onlyKeeperAndAbove

	uint256_usdrAmount	
setUsdrAmounts	address_vault address[] memory_tokens uint256[] memory_usdrAmounts	onlyKeeperAndAbove
updateUsdrSupply	uint256_usdrAmount	onlyKeeperAndAbove
setShortsTrackerAveragePriceWeight	uint256_shortsTrackerAveragePriceWeight	onlyAdmin
setLpCooldownDuration	uint256_cooldownDuration	onlyAdmin
removeAdmin	address_token address_account	onlyAdmin
setIsSwapEnabled	address_vault bool_isSwapEnabled	onlyKeeperAndAbove
setTier	address_referralStorage uint256_tierId uint256_totalRebate uint256_discountShare	onlyKeeperAndAbove
setReferrerTier	address_referralStorage address_referrer uint256_tierId	onlyKeeperAndAbove
govSetCodeOwner	address_referralStorage bytes32_code address_newAccount	onlyKeeperAndAbove
setMaxGasPrice	address_vault uint256_maxGasPrice	onlyAdmin
withdrawFees	address_vault address_token address_receiver	onlyAdmin
batchWithdrawFees	address_vault address[] memory_tokens	onlyKeeperAndAbove
setInPrivateLiquidationMode	address_vault bool_inPrivateLiquidationMode	onlyAdmin
setLiquidator	address_vault address_liquidator bool_isActive	onlyAdmin

setInPrivateTransferMode	address_token bool_inPrivateTransferMode	onlyAdmin
batchSetBonusRewards	address_vester address[] memory_accounts uint256[] memory_amounts	onlyKeeperAndAbove
transferIn	address_sender address_token uint256_amount	onlyAdmin
signalApprove	address_token address_spender uint256_amount	onlyAdmin
approve	address_token address_spender uint256_amount	onlyAdmin
signalWithdrawToken	address_target address_token address_receiver uint256_amount	onlyAdmin
withdrawToken	address_target address_token address_receiver uint256_amount	onlyAdmin
signalMint	address_token address_receiver uint256_amount	onlyAdmin
processMint	address_token address_receiver uint256_amount	onlyAdmin
signalSetGov	address_target address_gov	onlyAdmin
setGov	address_target address_gov	onlyAdmin
signalSetHandler	address_target address_handler bool_isActive	onlyAdmin
setHandler	address_target	onlyAdmin

	address_handler bool_isActive	
signalSetPriceFeed	address_vault address_priceFeed	onlyAdmin
setPriceFeed	address_vault address_priceFeed	onlyAdmin
signalRedeemUsdr	address_vault address_token u int256_amount	onlyAdmin
redeemUsdr	address_vault a ddress_token uint256_amount	onlyAdmin
signalVaultSetTokenC onfig	address_vault address_token uint256_tokenDecimals uint256_tokenWeight uint256_minProfitBps uint256_maxUsdrAmount bool_isStable bool_isShortable	onlyAdmin
vaultSetTokenConfig	address_vault address_token uint256_tokenDecimals uint256_tokenWeight uint256_minProfitBps uint256_maxUsdrAmount bool_isStable bool_isShortable	onlyAdmin
cancelAction	bytes32_action	onlyAdmin
_mint	address_token address_receiver uint256_amount	private
_setPendingAction	bytes32_action	private
_validateAction	bytes32_action	private
_clearAction	bytes32_action	private

## Governable Contract

Name	Parameter	Attributes
setGov	address_gov	onlyGov

## Reader Contract

Name	Parameter	Attributes
setConfig	bool_hasMaxGlobalShortSizes	onlyGov
getMaxAmountIn	IVault_vault address_tokenIn address_tokenOut	public
getAmountOut	IVault_vault address_tokenIn address_tokenOut uint256_amountIn	public
getFeeBasisPoints	IVault_vault address_tokenIn address_tokenOut uint256_amountIn	public
getFees	address_vault address[] memory_tokens	public
getTotalStaked	address[] memory_yieldTokens	public
getStakingInfo	address_account address[] memory_yieldTrackers	public
getFundingRates	address_vault address_weth address[] memory_tokens	public
getTokenSupply	IERC20_token address[] memory_excludedAccounts	public
getTotalBalance	IERC20_token address[] memory_accounts	public



getTokenBalances	address_account address[] memory_tokens	public
getTokenBalancesWithSupplies	address_account address[] memory_tokens	public
getPrices	IVaultPriceFeed_priceFeed address[] memory_tokens	public
getVaultTokenInfo	address_vault address_weth uint256_usdrAmount address[] memory_tokens	public
getFullVaultTokenInfo	address_vault address_weth uint256_usdrAmount address[] memory_tokens	public
getPositions	address_vault address_account address[] memory_collateralTokens address[] memory_indexTokens bool[] memory_isLong	public

### VaultReader Contract

Name	Parameter	Attributes
getVaultTokenInfoV3	address_vault address_positionManage address_weth uint256_usdrAmount address[] memory_tokens	public view
getVaultTokenInfoV4	address_vault address_positionManager address_weth uint256_usdrAmount address[] memory_tokens	public view

### BalanceUpdater Contract

Name	Parameter	Attributes
updateBalance	address_vault address_token address_usdr uint256_usdrAmount	public

### BatchSender Contract

Name	Parameter	Attributes
setHandler	address_handler bool_isActive	onlyGov
send	IERC20_token address[] memory_accounts uint256[] memory_amounts	onlyHandler
sendAndEmit	IERC20_token address[] memory_accounts uint256[] memory_amounts uint256_typeId	onlyHandler
_send	IERC20_token address[] memory_accounts uint256[] memory_amounts uint256_typeId	private

## RewardReader Contract

Name	Parameter	Attributes
getDepositBalances	address _account address[] memory depositToken address[] memory _rewardTrackers	public view
getStakingInfo	address _account address[] memory _rewardTrackers	public view
getVestingInfoV2	address _account address[] memory _vesters	public view

## OrderBookReader Contract

Name	Parameter	Attributes
getIncreaseOrders	address payable _orderBookAddress address _account uint256[] memory _indices	external
getDecreaseOrders	address payable _orderBookAddress address _account uint256[] memory _indices	external
getSwapOrders	address payable _orderBookAddress address _account uint256[] memory _indices	external

### DexV3Aggregator Contract

Name	Parameter	Attributes
addPriceSource	address_source uint256_weight address[] memory_path	onlyGov
removePriceSource	address_source	onlyGov
calcPrice	none	public
latestAnswer	none	public
latestTimestamp	none	public
latestRound	none	public
getAnswer	uint256	public
getTimestamp	uint256_roundId	public
getRoundData	uint80_roundId	external
latestRoundData	none	external
description	none	external

### FastPriceEvents Contract

Name	Parameter	Attributes
setIsPriceFeed	address_priceFeed bool_isPriceFeed	onlyGov
emitPriceEvent	address_token uint256_price	external

### CustomV3Aggregator Contract

Name	Parameter	Attributes
setFastPriceFeed	address_priceFeed	onlyGov
setUpdater	address_updater bool_status	onlyGov
updateAnswer	int256_answer	onlyUpdater
updateRoundData	uint80_roundId int256_answer uint256_timestamp uint256_startedAt	onlyUpdater
getRoundData	uint80_roundId	external
latestRoundData	none	external
description	none	external

### FastPriceFeed Contract

Name	Parameter	Attributes
initialize	uint256_minAuthorizations address[] memory_signers address[] memory_updaters	onlyGov
setSigner	address_account bool_isActive	onlyGov
setUpdater	address_account bool_isActive	onlyGov
setFastPriceEvents	address_fastPriceEvents	onlyGov
setVaultPriceFeed	address_vaultPriceFeed	onlyGov
setMaxTimeDeviation	uint256_maxTimeDeviation	onlyGov
setPriceDuration	uint256_priceDuration	onlyGov
setMaxPriceUpdateDelay	uint256_maxPriceUpdateDelay	onlyGov

setSpreadBasisPoint IfInactive	uint256 _spreadBasisPointsIfInactive	onlyGov
setSpreadBasisPoint IfChainError	uint256 _spreadBasisPointsIfChainError	onlyGov
setMinBlockInterval	uint256 _minBlockInterval	onlyGov
setIsSpreadEnabled	bool _isSpreadEnabled	onlyGov
setLastUpdatedAt	uint256 _lastUpdatedAt	onlyGov
setMaxDeviationBasi sPoints	uint256 _maxDeviationBasisPoints	onlyGov
setMaxCumulativeDe ltaDiffs	address[] memory _tokens uint256[] memory _maxCumulativeDeltaDiffs	onlyGov
setPriceDataInterval	uint256 _priceDataInterval	onlyGov
setMinAuthorization s	uint256 _minAuthorizations	onlyGov
setTokens	address[] memory _tokens uint256[] memory _tokenPrecisions	onlyGov
setPrices	address[] memory _tokens uint256[] memory _prices uint256 _timestamp	onlyUpdate r
setCompactedPrices	uint256[] memory _priceBitArray uint256 _timestamp	onlyUpdate r
setPricesWithBits	uint256 _priceBits uint256 _timestamp	onlyUpdate r
setPricesWithBitsAn dExecute	uint256 _priceBits, uint256 _timestamp uint256 _endIndexForIncreasePositions uint256 _endIndexForDecreasePositions uint256 _maxIncreasePositions, uint256 _maxDecreasePositions	onlyUpdate r
disableFastPrice	none	onlySigner
enableFastPrice	none	onlySigner
getPrice	address _token uint256 _refPrice bool _maximise	public
favorFastPrice	address _token	public
getPriceData	address _token	public

_setPricesWithBits	uint256 _priceBits uint256 _timestamp	private
_setPrice	address _token uint256 _price address _vaultPriceFeed address _fastPriceEvents	private
_setPriceData	address _token uint256 _refPrice uint256 _cumulativeRefDelta uint256 _cumulativeFastDelta	private
_emitPriceEvent	address _fastPriceEvents address _token uint256 _price	private
_setLastUpdatedValues	uint256 _timestamp	private

### LpManager Contract

Name	Parameter	Attributes
setInPrivateMode	bool _inPrivateMode	onlyGov
setShortsTracker	IShortsTracker _shortsTracker	onlyGov
setShortsTrackerAveragePriceWeight	uint256 _shortsTrackerAveragePriceWeight	onlyGov
setHandler	address _handler bool _isActive	onlyGov
setCooldownDuration	uint256 _cooldownDuration	onlyGov
setAumAdjustment	uint256 _aumAddition uint256 _aumDeduction	onlyGov
addLiquidity	address _token uint256 _amount uint256 _minUsdr uint256 _minLp	external
addLiquidityForAccount	address _fundingAccount	external

	address_account address_token uint256_amount uint256_minUsdr uint256_minLp	
removeLiquidity	address_tokenOut uint256_lpAmount uint256_minOut address_receiver	external
removeLiquidityForAccount	address_account address_tokenOut uint256_lpAmount uint256_minOut address_receiver	external
getPrice	bool_maximise	external
getAums	none	public
getAumInUsdr	bool maximise	public
getAum	bool maximise	public
getGlobalShortDelta	address_token uint256_price uint256_size	public
getGlobalShortAveragePrice	address_token	public
_addLiquidity	address_fundingAccount address_account address_token uint256_amount uint256_minUsdr uint256_minLp	private
_removeLiquidity	address_account address_tokenOut uint256_lpAmount uint256_minOut address_receiver	private
_validateHandler	none	private



## VaultWrapper Contract

Name	Parameter	Attributes
setShouldToggleIsLeverageEnabled	bool _shouldToggleIsLeverageEnabled	onlyGov
setMarginFeeBasisPoints	uint256 _marginFeeBasisPoints uint256 _maxMarginFeeBasisPoints	onlyGov
enableLeverage	address _vault	external
disableLeverage	address _vault	external

## PositionRouter Contract

Name	Parameter	Attributes
setPositionKeeper	address _account bool _isActive	onlyAdmin
setCallbackGasLimit	uint256 _callbackGasLimit	onlyAdmin
setMinExecutionFee	uint256 _minExecutionFee	onlyAdmin
setIsLeverageEnabled	bool _isLeverageEnabled	onlyAdmin
setDelayValues	uint256 _minBlockDelayKeeper uint256 _minTimeDelayPublic uint256 _maxTimeDelay	onlyAdmin
setRequestKeysStartValues	uint256 _increasePositionRequestKeysStart uint256 _decreasePositionRequestKeysStart	onlyAdmin
executeIncreasePositions	uint256 _endIndex address payable _executionFeeReceiver	onlyPosition Keeper
executeDecreasePositions	uint256 _endIndex address payable _executionFeeReceiver	onlyPosition Keeper
createIncreasePosition	address[] memory _path address _indexToken uint256 _amountIn	external

	uint256_minOut uint256_sizeDelta bool_isLong uint256_acceptablePrice uint256_executionFee bytes32_referralCode address_callbackTarget	
createIncreasePositionETH	address[] memory_path address_indexToken uint256_minOut uint256_sizeDelta bool_isLong uint256_acceptablePrice uint256_executionFee bytes32_referralCode address_callbackTarget	external
createDecreasePosition	address[] memory_path address_indexToken uint256_collateralDelta uint256_sizeDelta bool_isLong address_receiver uint256_acceptablePrice uint256_minOut uint256_executionFee bool_withdrawETH address_callbackTarget	external
getRequestQueueLengths	none	external
executeIncreasePosition	bytes32_key address payable_executionFeeReceiver	public
cancelIncreasePosition	bytes32_key address payable_executionFeeReceiver	public
executeDecreasePosition	bytes32_key address payable_executionFeeReceiver	public
cancelDecreasePosition	bytes32_key address payable_executionFeeReceiver	public

getRequestKey	address_account uint256_index	public
getIncreasePositionRequestPath	bytes32_key	public
getDecreasePositionRequestPath	bytes32_key	public
_setTraderReferralCode	bytes32_referralCode	internal
_validateExecution	uint256_positionBlockNumber uint256_positionBlockTime address_account	internal
_validateCancellation	uint256_positionBlockNumber uint256_positionBlockTime address_account	internal
_createIncreasePosition	address_account address[] memory_path address_indexToken uint256_amountIn uint256_minOut uint256_sizeDelta bool_isLong uint256_acceptablePrice uint256_executionFee bool_hasCollateralInETH address_callbackTarget	internal
_storeIncreasePositionRequest	IncreasePositionRequest memory_request	internal
_storeDecreasePositionRequest	DecreasePositionRequest memory_request	internal
_createDecreasePosition	address_account address[] memory_path address_indexToken uint256_collateralDelta uint256_sizeDelta bool_isLong address_receiver uint256_acceptablePrice	internal

	uint256_minOut	
	uint256_executionFee	
	bool_withdrawETH	
	address_callbackTarget	
_callRequestCallback	address_callbackTarget	
	bytes32_key	internal
	bool_wasExecuted	
	bool_isIncrease	

### VaultPriceFeed Contract

Name	Parameter	Attributes
setGov	address_gov	onlyGov
setChainlinkFlags	address_chainlinkFlags	onlyGov
setAdjustment	address_token bool_isAdditive uint256_adjustmentBps	onlyGov
setUseV2Pricing	bool_useV2Pricing	onlyGov
setIsAmmEnabled	bool_isEnabled	onlyGov
setIsSecondaryPriceEnabled	bool_isEnabled	onlyGov
setSecondaryPriceFeed	address_secondaryPriceFeed	onlyGov
setTokens	address_btc address_eth address_bnb	onlyGov
setPairs	address_bnbBusd address_ethBnb address_btcBnb	onlyGov
setSpreadBasisPoints	address_token uint256_spreadBasisPoints	onlyGov
setSpreadThresholdBasisPoints	uint256_spreadThresholdBasisPoints	onlyGov
setFavorPrimaryPrice	bool_favorPrimaryPrice	onlyGov

setPriceSampleSpace	uint256_priceSampleSpace	onlyGov
setMaxStrictPriceDeviation	uint256_maxStrictPriceDeviation	onlyGov
setTokenConfig	address_token address_priceFeed uint256_priceDecimals bool_isStrictStable	onlyGov
getPrice	address_token bool_maximise bool_includeAmmPrice bool /*_useSwapPricing*/	public override view
getPriceV1	address_token bool_maximise bool_includeAmmPrice	public view
getPriceV2	address_token bool_maximise bool_includeAmmPrice	public view
getAmmPriceV2	address_token bool_maximise uint256_primaryPrice	public view
getLatestPrimaryPrice	address_token	public override view
getPrimaryPrice	address_token bool_maximise	public override view
getSecondaryPrice	address_token uint256_referencePrice bool_maximise	public view
getAmmPrice	address_token	public override view
getPairPrice	address_pair bool_divByReserve0	public view

## PositionManager Contract

Name	Parameter	Attributes
setOrderKeeper	address _account bool _isActive	onlyAdmin
setLiquidator	address _account bool _isActive	onlyAdmin
setPartner	address _account bool _isActive	onlyAdmin
setShouldValidateIncreaseOrder	bool _shouldValidateIncreaseOrder	onlyAdmin
increasePosition	address[] memory _path address _indexToken uint256 _amountIn uint256 _minOut uint256 _sizeDelta bool _isLong uint256 _price	onlyPartners
increasePositionETH	address[] memory _path address _indexToken uint256 _minOut uint256 _sizeDelta bool _isLong uint256 _price	onlyPartners
decreasePosition	address _collateralToken address _indexToken uint256 _collateralDelta uint256 _sizeDelta bool _isLong address _receiver uint256 _price	onlyPartners
decreasePositionETH	address _collateralToken address _indexToken uint256 _collateralDelta uint256 _sizeDelta bool _isLong	onlyPartner

	address_receiver uint256_price	
decreasePositionAndSwap	address[] memory_path address_indexToken uint256_collateralDelta uint256_sizeDelta bool_isLong address_receiver uint256_price uint256_minOut	onlyPartner
decreasePositionAndSwapETH	address[] memory_path address_indexToken uint256_collateralDelta uint256_sizeDelta bool_isLong address_receiver uint256_price uint256_minOut	onlyPartner
liquidatePosition	address_account address_collateralToken address_indexToken bool_isLong,address_feeReceiver	onlyLiquidator
executeSwapOrder	address_account uint256_orderIndex address_payable_feeReceiver	onlyOrderKeeper
executeIncreaseOrder	address_account uint256_orderIndex address_payable_feeReceiver	onlyOrderKeeper
executeDecreaseOrder	address_account uint256_orderIndex address_payable_feeReceiver	onlyOrderKeeper
_validateIncreaseOrder	address_account uint256_orderIndex	internal view

## ShortsTracker Contract

Name	Parameter	Attributes
setHandler	address_handler bool_isActive	onlyGov
_setGlobalShortAveragePrice	address_token uint256_averagePrice	internal
setIsGlobalShortDataReady	bool value	onlyGov
updateGlobalShortData	address_account address_collateralToken address_indexToken bool_isLong uint256_sizeDelta uint256_markPrice bool_isIncrease	onlyHandler
getGlobalShortDelta	address_token	public
setInitData	address[] calldata_tokens uint256[] calldata_averagePrices	onlyGov
getNextGlobalShortData	address_account address_collateralToken address_indexToken uint256_nextPrice uint256_sizeDelta bool_isIncrease	public
getRealisedPnl	address_account address_collateralToken address_indexToken uint256_sizeDelta bool_isIncrease	public
_getNextGlobalAveragePrice	uint256_averagePrice uint256_nextPrice uint256_nextSize uint256_delta int256_realisedPnl	public
_getNextDelta	uint256_delta uint256_averagePrice	internal



uint256 \_nextPrice  
int256 \_realisedPnl

### OrderBook Contract

Name	Parameter	Attributes
initialize	address _router address _vault address _weth address _usdr uint256 _minExecutionFee uint256 _minPurchaseTokenAmountUsd	onlyGov
setMinExecutionFee	uint256 _minExecutionFee	onlyGov
setMinPurchaseTokenAmountUsd	uint256 _minPurchaseTokenAmountUsd	onlyGov
setGov	address _gov	onlyGov
getSwapOrder	address _account uint256 _orderIndex	public
createSwapOrder	address[] memory _path uint256 _amountIn uint256 _minOut uint256 _triggerRatio bool _triggerAboveThreshold uint256 _executionFee bool _shouldWrap bool _shouldUnwrap	external
_createSwapOrder	address _account address[] memory _path uint256 _amountIn uint256 _minOut uint256 _triggerRatio bool _triggerAboveThreshold bool _shouldUnwrap	private

	uint256_executionFee	
cancelMultiple	uint256[] memory_swapOrderIndexes uint256[] memory_increaseOrderIndexes uint256[] memory_decreaseOrderIndexes	external
cancelSwapOrder	uint256_orderIndex	public
getUsdrMinPrice	address_otherToken	public
validateSwapOrderPriceWithTriggerAboveThreshold	address[] memory_path uint256_triggerRatio	public
updateSwapOrder	uint256_orderIndex uint256_minOut uint256_triggerRatio bool_triggerAboveThreshold	external
executeSwapOrder	address_account uint256_orderIndex address payable_feeReceiver	external
validatePositionOrderPrice	bool_triggerAboveThreshold uint256_triggerPrice address_indexToken bool_maximizePrice bool_raise	public
getDecreaseOrder	address_account uint256_orderIndex	public
getIncreaseOrder	address_account uint256_orderIndex	public
createIncreaseOrder	address[] memory_path uint256_amountIn address_indexToken uint256_minOut uint256_sizeDelta address_collateralToken bool_isLong uint256_triggerPrice bool_triggerAboveThreshold uint256_executionFee bool_shouldWrap	external
_createIncreaseOrder	address_account	private

	address _purchaseToken uint256 _purchaseTokenAmount address _collateralToken address _indexToken uint256 _sizeDelta bool _isLong uint256 _triggerPrice bool _triggerAboveThreshold uint256 _executionFee	
updateIncreaseOrder	uint256 _orderIndex uint256 _sizeDelta uint256 _triggerPrice bool _triggerAboveThreshold	external
cancelIncreaseOrder	uint256 _orderIndex	public
executeIncreaseOrder	address _address uint256 _orderIndex address payable _feeReceiver	external
createDecreaseOrder	address _indexToken uint256 _sizeDelta address _collateralToken uint256 _collateralDelta bool _isLong uint256 _triggerPrice bool _triggerAboveThreshold	external
_createDecreaseOrder	address _account address _collateralToken uint256 _collateralDelta address _indexToken uint256 _sizeDelta bool _isLong uint256 _triggerPrice bool _triggerAboveThreshold	private
executeDecreaseOrder	address _address uint256 _orderIndex address payable _feeReceiver	external
cancelDecreaseOrder	uint256 _orderIndex	public
updateDecreaseOrder	uint256 _orderIndex	external

	uint256 _collateralDelta uint256 _sizeDelta uint256 _triggerPrice bool _triggerAboveThreshold	
_transferInETH	none	private
_transferOutETH	uint256 _amountOut address payable _receiver	private
_swap	none	
_vaultSwap	address _tokenIn address _tokenOut uint256 _minOut address _receiver	private

### Vault Contract

Name	Parameter	Attributes
initialize	address _router address _usdr address _priceFeed uint256 _liquidationFeeUsd uint256 _fundingRateFactor uint256 _stableFundingRateFactor	external
allWhitelistedTokensLength	none	external
setInManagerMode	bool _inManagerMode	external
setManager	address _manager bool _isManager	external
setInPrivateLiquidationMode	bool _inPrivateLiquidationMode	external
setLiquidator	address _liquidator bool _isActive	external
setIsSwapEnabled	bool _isSwapEnabled	external
setIsLeverageEnabled	bool _isLeverageEnabled	external

setMaxGasPrice	uint256_maxGasPrice	external
setWrapper	address_wrapper	external
setGov	address_gov	external
setPriceFeed	address_priceFeed	external
setMaxLeverage	uint256_maxLeverage	external
setBufferAmount	address_token uint256_amount	external
setFees	uint256_taxBasisPoints uint256_stableTaxBasisPoints uint256_mintBurnFeeBasisPoints uint256_swapFeeBasisPoints uint256_stableSwapFeeBasisPoints uint256_marginFeeBasisPoints uint256_liquidationFeeUsd uint256_minProfitTime bool_hasDynamicFees	external
setFundingRate	uint256_fundingInterval uint256_fundingRateFactor uint256_stableFundingRateFactor	external
setTokenConfig	address_token uint256_tokenDecimals uint256_tokenWeight uint256_minProfitBps uint256_maxUsdrAmount bool_isStable bool_isShortable	external
clearTokenConfig	address_token	external
withdrawFees	address_token address_receiver	external
addRouter	address_router	external
removeRouter	address_router	external
setUsdrAmount	address_token uint256_amount	external
upgradeVault	address_newVault address_token uint256_amount	external

directPoolDeposit	address_token	external
buyUSDR	address_token address_receiver	external
sellUSDR	address_token address_receiver	external
swap	address_tokenIn address_tokenOut address_receiver	external
increasePosition	address_account address_collateralToken address_indexToken uint256_sizeDelta bool_isLong	external
decreasePosition	address_account address_collateralToken address_indexToken uint256_collateralDelta uint256_sizeDelta bool_isLong address_receiver	external
_decreasePosition	address_account address_collateralToken address_indexToken uint256_collateralDelta uint256_sizeDelta bool_isLong address_receiver	private
liquidatePosition	address_account address_collateralToken address_indexToken bool_isLong address_feeReceiver	external
validateLiquidation	address_account address_collateralToken address_indexToken bool_isLong bool_raise	public

getMaxPrice	address_token	public
getMinPrice	address_token	public
getRedemptionAmount	address_token uint256_usdrAmount	public
getRedemptionCollateral	address_token	public
getRedemptionCollateralUsd	address_token	public
adjustForDecimals	uint256_amount address_tokenDiv address_tokenMul	public
tokenToUsdMin	address_token uint256_tokenAmount	public
usdToTokenMax	address_token uint256_usdAmount	public
usdToTokenMin	address_token uint256_usdAmount	public
usdToToken	address_token uint256_usdAmount uint256_price	public
getPosition	address_account address_collateralToken address_indexToken bool_isLong	public
getPositionKey	address_account address_collateralToken address_indexToken bool_isLong	public
updateCumulativeFundingRate	address_token	public
getNextFundingRate	address_token	public
getUtilisation	address_token	public
getPositionLeverage	address_account address_collateralToken address_indexToken bool_isLong	public

getNextAveragePrice	address_indexToken uint256_size uint256_averagePrice bool_isLong uint256_nextPrice uint256_sizeDelta uint256_lastIncreasedTime	public
getNextGlobalShortAveragePrice	address_indexToken uint256_nextPrice uint256_sizeDelta	public
getGlobalShortDelta	address_token	public
getPositionDelta	address_account address_collateralToken address_indexToken bool_isLong	public
getDelta	address_indexToken uint256_size uint256_averagePrice bool_isLong uint256_lastIncreasedTime	public
getFundingFee	address_token uint256_size uint256_entryFundingRate	public
getPositionFee	uint256_sizeDelta	public
getFeeBasisPoints	address_token uint256_usdrDelta uint256_feeBasisPoints uint256_taxBasisPoints bool_increment	public
getTargetUsdrAmount	address_token	public
_reduceCollateral	address_account address_collateralToken address_indexToken uint256_collateralDelta uint256_sizeDelta bool_isLong	private



_validatePosition	uint256_size uint256_collateral	private
_validateRouter	address_account	private
_validateTokens	address_collateralToken address_indexToken bool_isLong	private
_collectSwapFees	address_token uint256_amount uint256_feeBasisPoints	private
_collectMarginFees	address_token uint256_sizeDelta uint256_size uint256_entryFundingRate	private
_transferIn	address_token	private
_transferOut	address_token uint256_amount address_receiver	private
_updateTokenBalance	address_token	private
_increasePoolAmount	address_token uint256_amount	private
_decreasePoolAmount	address_token uint256_amount	private
_validateBufferAmount	address_token	private
_increaseUsdrAmount	address_token uint256_amount	private
_decreaseUsdrAmount	address_token uint256_amount	private
_increaseReservedAmount	address_token uint256_amount	private
_decreaseReservedAmount	address_token uint256_amount	private
_increaseGuaranteedUsd	address_token uint256_usdAmount	private
_decreaseGuaranteedUsd	address_token uint256_usdAmount	private

_decreaseGlobalShortSize	address_token uint256_amount	private
_onlyGov	none	private
_validateManager	none	private
_validateGasPrice	none	private
_onlyGovOrWrapper	none	private

### Router Contract

Name	Parameter	Attributes
setGov	address_gov	external
addPlugin	address_plugin	external
removePlugin	address_plugin	external
approvePlugin	address_plugin	external
denyPlugin	address_plugin	external
pluginTransfer	address_token address_account address_receiver uint256_amount	external
pluginIncreasePosition	address_account address_collateralToken address_indexToken uint256_sizeDelta bool_isLong	external
pluginDecreasePosition	address_account address_collateralToken address_indexToken uint256_collateralDelta uint256_sizeDelta bool_isLong address_receiver	external
directPoolDeposit	address_token	external

	uint256 _amount	
swap	address[] memory _path uint256 _amountIn uint256 _minOut address _receiver	public
swapETHToTokens	address[] memory _path uint256 _minOut address _receiver	external
swapTokensToETH	address[] memory _path uint256 _amountIn uint256 _minOut address payable _receiver	external
increasePosition	address[] memory _path address _indexToken uint256 _amountIn uint256 _minOut uint256 _sizeDelta bool _isLong uint256 _price	external
increasePositionETH	address[] memory _path address _indexToken uint256 _minOut uint256 _sizeDelta bool _isLong uint256 _price	external
decreasePosition	address _collateralToken address _indexToken uint256 _collateralDelta uint256 _sizeDelta bool _isLong address _receiver uint256 _price	external
decreasePositionETH	address _collateralToken address _indexToken uint256 _collateralDelta uint256 _sizeDelta bool _isLong	external

	address payable _receiver uint256 _price	
decreasePositionAndSwap	address[] memory _path address _indexToken uint256 _collateralDelta uint256 _sizeDelta bool _isLong address _receiver uint256 _price uint256 _minOut	external
decreasePositionAndSwapETH	address[] memory _path address _indexToken uint256 _collateralDelta uint256 _sizeDelta bool _isLong address payable _receiver uint256 _price uint256 _minOut	external
_increasePosition	address _collateralToken address _indexToken uint256 _sizeDelta bool _isLong uint256 _price	private
_decreasePosition	address _collateralToken address _indexToken uint256 _collateralDelta uint256 _sizeDelta bool _isLong address _receiver uint256 _price	private
_transferETHToVault	none	private
_transferOutETH	uint256 _amountOut address payable _receiver	private
_swap	address[] memory _path uint256 _minOut address _receiver	private
_vaultSwap	address _tokenIn	private

	address_tokenOut	
	uint256_minOut	
	address_receiver	
_sender	none	private
_validatePlugin	address_account	private

### BasePositionManager Contract

Name	Parameter	Attributes
setAdmin	address_admin	onlyGov
setDepositFee	uint256_depositFee	onlyAdmin
setIncreasePositionBufferBps	uint256_increasePositionBufferBps	onlyAdmin
setReferralStorage	address_referralStorage	onlyAdmin
setMaxGlobalSizes	address[] memory_tokens uint256[] memory_longSizes uint256[] memory_shortSizes	onlyAdmin
withdrawFees	address_token address_receiver	onlyAdmin
approve	address_token address_spender uint256_amount	onlyGov
sendValue	addresspayable_receiver uint256_amount	onlyGov
_validateMaxGlobalSize	address_indexToken bool_isLong uint256_sizeDelta	internal
_increasePosition	address_account address_collateralToken address_indexToken uint256_sizeDelta bool_isLong uint256_price	internal

_decreasePosition	address_account address_collateralToken address_indexToken uint256_collateralDelta uint256_sizeDelta bool_isLong address_receiver uint256_price	internal
_emitIncreasePosition Referral	address_account uint256_sizeDelta	internal
_emitDecreasePosition Referral	address_account uint256_sizeDelta	internal
_swap	address[] memory_path uint256_minOut address_receiver	internal
_vaultSwap	address_tokenIn address_tokenOut uint256_minOut address_receiver	internal
_transferInETH	none	internal
_transferOutETHWith GasLimitIgnoreFail	uint256_amountOut address_payable_receiver	internal
_collectFees	address_account address[] memory_path uint256_amountIn address_indexToken bool_isLong uint256_sizeDelta	internal
_shouldDeductFee	address_account address[] memory_path uint256_amountIn address_indexToken bool_isLong uint256_sizeDelta	internal

## RewardTracker Contract

Name	Parameter	Attributes
initialize	address[] memory _depositTokens address _distributor	onlyGov
setDepositToken	address _depositToken bool _isDepositToken	onlyGov
setInPrivateTransferMode	bool _inPrivateTransferMode	onlyGov
setInPrivateStakingMode	bool _inPrivateStakingMode	onlyGov
setInPrivateClaimingMode	bool _inPrivateClaimingMode	onlyGov
setHandler	address _handler bool _isActive	onlyGov
withdrawToken	address _token address _account uint256 _amount	onlyGov
balanceOf	address _account	external
stake	address _depositToken uint256 _amount	external
stakeForAccount	address _fundingAccount address _account address _depositToken uint256 _amount	external
unstake	address _depositToken uint256 _amount	external
unstakeForAccount	address _account address _depositToken uint256 _amount address _receiver	external
transfer	address _recipient uint256 _amount	external
allowance	address _owner address _spender	external
approve	address _spender	external

	uint256 _amount	
transferFrom	address _sender address _recipient uint256 _amount	external
tokensPerInterval	none	external
updateRewards	none	external
claim	address _receiver	external
claimForAccount	address _account address _receiver	external
claimable	address _account	public
rewardToken	none	public
_claim	address _account address _receiver	private
_mint	address _account uint256 _amount	internal
_burn	address _account uint256 _amount	internal
_transfer	address _sender address _recipient uint256 _amount	private
_approve	address _owner address _spender uint256 _amount	private
_validateHandler	none	private
_stake	address _fundingAccount address _account address _depositToken uint256 _amount	private
_unstake	address _account address _depositToken uint256 _amount address _receiver	private
_updateRewards	address _account	private



## RewardRouterV1 Contract

Name	Parameter	Attributes
initialize	address_weth address_lp address_feeLpTracker address_lpManager	onlyGov
withdrawToken	address_token address_account uint256_amount	onlyGov
mintAndStakeLp	address_token uint256_amount uint256_minUsdr uint256_minLp	external
mintAndStakeLpETH	uint256_minUsdr uint256_minLp	external
unstakeAndRedeemLp	address_tokenOut uint256_lpAmount uint256_minOut address_receiver	external
unstakeAndRedeemLp ETH	uint256_lpAmount uint256_minOut addresspayable_receiver	external
claim	none	external
claimFees	none	external
handleRewards	bool_shouldConvertWethToEth	external
signalTransfer	address_receiver	external
acceptTransfer	address_sender	external
_validateReceiver	address_receiver	private

## RewardDistributor Contract

Name	Parameter	Attributes
setAdmin	address_admin	onlyGov
withdrawToken	address_token address_account uint256_amount	onlyGov
updateLastDistributionTime	none	onlyAdmin
setTokensPerInterval	uint256_amount	onlyAdmin
pendingRewards	none	public
distribute	none	external

## ReferralStorage Contract

Name	Parameter	Attributes
setHandler	address_handler bool_isActive	onlyGov
setTier	uint256_tierId uint256_totalRebate uint256_discountShare	onlyGov
setReferrerTier	address_referrer uint256_tierId	onlyGov
setReferrerDiscountShare	uint256_discountShare	external
setTraderReferralCode	address_account bytes32_code	onlyHandler
setTraderReferralCodeByUser	bytes32_code	external
registerCode	bytes32_code	external
setCodeOwner	bytes32_code address_newAccount	external
govSetCodeOwner	bytes32_code	onlyGov

	address_newAccount	
getTraderReferralInfo	address_account	external
_setTraderReferralCode	address_account bytes32_code	private

### ReferralReader Contract

Name	Parameter	Attributes
getCodeOwners	IReferralStorage_referralStorage bytes32[] memory_codes	public

## Router Contract

Name	Parameter	Attributes
setESBT	address_esbt	onlyOwner
setValidateContract	bool_valid	onlyOwner
setInfoCenter	address_infCenter	onlyOwner
addPlugin	address_plugin	onlyOwner
removePlugin	address_plugin	onlyOwner
withdrawToken	address_account address_token uint256_amount	onlyOwner
approvePlugin	address_plugin	external
denyPlugin	address_plugin	external
pluginTransfer	address_token address_account address_receiver uint256_amount	external
pluginIncreasePosition	address_account address_collateralToken address_indexToken uint256_sizeDelta bool_isLong	external
pluginDecreasePosition	address_account address_collateralToken address_indexToken uint256_collateralDelta uint256_sizeDelta bool_isLong address_receiver	external
directPoolDeposit	address_token uint256_amount	external
decreasePosition	address_collateralToken address_indexToken uint256_collateralDelta	external

	uint256 _sizeDelta bool _isLong address _receiver uint256 _price	
decreasePositionETH	address _collateralToken address _indexToken uint256 _collateralDelta uint256 _sizeDelta bool _isLong address payable _receiver uint256 _price	external
_increasePosition	address _collateralToken address _indexToken uint256 _sizeDelta bool _isLong uint256 _price	private
_decreasePosition	address _collateralToken address _indexToken uint256 _collateralDelta uint256 _sizeDelta bool _isLong address _receiver uint256 _price	private
_transferETHToVault	none	private
_transferOutETH	uint256 _amountOut address payable _receiver	private
_vaultSwap	address _tokenIn address _tokenOut uint256 _minOut address _receiver	private
_sender	none	private
_validatePlugin	address _account	private
isContract	address addr	private

## VaultPriceFeedV21Fast Contract

Name	Parameter	Attributes
adjustmentBasisPoints	address_token	external
isAdjustmentAdditive	address_token	external
setAdjustment	address_token bool_isAdditive uint256_adjustmentBps	external
setSpreadBasisPoints	address_token uint256_spreadBasisPoints	external
getOrigPrice	address_token	external
priceVariancePer1Million	address_token	external
getPrimaryPrice	address_token bool_maximise	external
increasePositionRequestKeysStart	none	external
decreasePositionRequestKeysStart	none	external
executeIncreasePositions	uint256_count address payable_executionFeeReceiver	external
executeDecreasePositions	uint256_count address payable_executionFeeReceiver	external
getRequestQueueLengths	none	external
setPriceMethod	uint8_setT	onlyOwner
setPriceVariance	uint256_priceVariance	onlyOwner
setSafePriceTimeGap	uint256_gap	onlyOwner
setAdjustment	address_token bool_isAdditive uint256_adjustmentBps	onlyOwner
setSpreadBasisPoints	address_token uint256_spreadBasisPoints	onlyOwner

_getCombPrice	address_token bool_maximise	internal
getOrigPrice	address_token	public
getChainlinkPrice	address_token bool_max	public
getPrimaryPrice	address_token bool_maximise	public
setUpdater	address_account bool_isActive	onlyOwner
setSignPrefixCode	address_updater uint256_setCode	onlyOwner
setTimeTolerance	uint256_tol	onlyOwner
setTokenChainlinkConfig	address_token address_chainlinkContract bool_isStrictStable	onlyOwner
addPositionRouter	address_positionRouter	onlyOwner
VerifyMessage	bytes32_hashedMessage uint8_v bytes32_r bytes32_s	public
splitSignature	bytes_sig	public
recoverSigner	bytes32_ethSignedMessageHash bytes_signature	public

## VaultUtils Contract

Name	Parameter	Attributes
priceVariancePer1Million	address_token	external
setMaxProfitRatio	uint256_setRatio	onlyOwner
setSpreadBasis	address_token uint256_spreadBasis uint256_maxSpreadBasis uint256_minSpreadCalUSD	onlyOwner
setMaxGlobalSize	address_token uint256_amountLong uint256_amountShort	onlyOwner
setTradingLimit	address_token uint256_maxShortSize uint256_maxLongSize uint256_maxSize uint256_maxRatio uint256_countMinSize	onlyOwner
setOnlyRouterSwap	bool_onlyRS	onlyOwner
setLiquidator	address_liquidator bool_isActive	onlyOwner
setInPrivateLiquidationMode	bool_inPrivateLiquidationMode	onlyOwner
setPremiumRate	uint256_premiumBasisPoints int256_posIndexMaxPoints int256_negIndexMaxPoints uint256_maxPremiumBasisErrorUSD	onlyOwner
setFundingRate	uint256_fundingRateFactor uint256_stableFundingRateFactor	onlyOwner
setMaxLeverage	uint256_maxLeverage	onlyOwner
setTaxRate	uint256_taxMax uint256_taxTime	onlyOwner



getLatestFundingRatePerSec	address_token	public
hRateToSecRate	uint256_comRate	public
hRateToSecRateInt	int256_comRate	public
getLatestLSRate	address_token	public
updateRate	address_token	public
getNextIncreaseTime	uint256_prev_time uint256_prev_size uint256_sizeDelta	public
validateIncreasePosition	address_collateralToken address_indexToken uint256_size uint256_sizeDelta bool_isLong	external
validateDecreasePosition	VaultMSData.Position_position uint256_sizeDelta uint256_collateralDelta	external
getPositionKey	address_account address_collateralToken address_indexToken bool_isLong uint256_keyID	public
getPositionInfo	address_account address_collateralToken address_indexToken bool_isLong	public
getPositionsInfo	uint256_start uint256_end	public
getNextAveragePrice	uint256_size uint256_averagePrice uint256_nextPrice uint256_sizeDelta bool_isIncrease	public
getInitialPosition	address_account address_collateralToken address_indexToken	public

	uint256 _sizeDelta bool _isLong uint256 _price	
getPositionNextAveragePrice	uint256 _size uint256 _averagePrice uint256 _nextPrice uint256 _sizeDelta bool _isIncrease	public
calculateTax	uint256 _profit uint256 _aveIncreaseTime	public
validateLiquidation	bytes32 _key bool _raise	public
validateLiquidationParameter	address _account address _collateralToken address _indexToken bool _isLong bool _raise	public
_validateLiquidation	VaultMSData.Position position bool _raise	public
getPositionImpactRatio	address _token uint256 _size	public
getImpactedPrice	address _token uint256 _sizeDelta uint256 _price bool _isLong	public
getFundingFee	VaultMSData.Position _position VaultMSData.TradingFee _tradingFee	public
getPremiumFee	VaultMSData.Position _position VaultMSData.TradingFee _tradingFee	public
getBuyUsdxFeeBasisPoints	address _token uint256 _usdxAmount	public
getSellUsdxFeeBasisPoints	address _token uint256 _usdxAmount	public
getSwapFeeBasisPoints	address _tokenIn address _tokenOut uint256 _usdxAmount	public

---

getFeeBasisPoints	address_token uint256_usdxDelta uint256_feeBasisPoints uint256_taxBasisPoints bool_increment	public
_validate	bool_condition uint256_errorCode	private
getTradingTax	address_token	public
getTradingLimit	address_token	public
tokenUtilization	address_token	public
getTargetUsdxAmount	address_token	public
validLiq	address_account	public

---

---

## 4. Audit details

### 4.1 Findings Summary

Severity	Found	Resolved	Acknowledged
● High	0	0	0
● Medium	0	0	0
● Low	1	0	1
● Info	8	2	6

## 4.2 Risk distribution

Name	Risk level	Repair status
Administrator Permissions	Low	Acknowledged
Same address judgment	Info	Acknowledged
Redundant codes	Info	Acknowledged
Logical Design Flaw	Info	Acknowledged
Reentry attack	No	normal
Variables are updated	No	normal
Floating Point and Numeric Precision	No	normal
Default visibility	No	normal
tx.origin authentication	No	normal
Faulty constructor	No	normal
Unverified return value	No	normal
Insecure random numbers	No	normal
Timestamp Dependent	No	normal
Transaction order dependency	No	normal
Delegatecall	No	normal
Call	No	normal
Denial of Service	No	normal
Fake recharge vulnerability	No	normal
Short address attack Vulnerability	No	normal
Uninitialized storage pointer	No	normal
Frozen account bypass	No	normal

---

Uninitialized	No	normal
Integer Overflow	No	normal

---

---

## 4.3 Risk audit details

### 4.3.1 Administrator permissions

- **Risk description**

The upgradeVault function is called for the gov privilege, when the gov privileged role is the EOA address, you can directly transfer funds out of the vault contract, it is recommended to use the TimeLock contract to restrict the operation of this function.

```
function upgradeVault(address _newVault, address _token, uint256 _amount) external {
    _onlyGov();
    IERC20(_token).safeTransfer(_newVault, _amount);
}
```

- **Safety advice**

Contract configuration related and important functions for high authority transfers try to use multi-signature or time lock control and avoid using EOA addresses for management.

- **Repair Status**

ROLLUP.FINANCE has Acknowledged.

### 4.3.2 Same address judgment

- Risk description

There are multiple contracts in the project with `_vaultSwap` functions, all of which are called from the vault contract. Since neither the `buyUSDR` function nor the `sellUSDR` function checks if the `_token` parameter is equal to the USDR token address, there may be a case where a transaction is executed with USDR to obtain USDR.

```
function _vaultSwap(address _tokenIn, address _tokenOut, uint256 _minOut, address _receiver) private returns (uint256) {
    uint256 amountOut;
    if (_tokenOut == rUSD) { // buyRUSD
        amountOut = IVault(vault).buyRUSD(_tokenIn, _receiver);
    } else if (_tokenIn == rUSD) { // sellRUSD
        amountOut = IVault(vault).sellRUSD(_tokenOut, _receiver);
    } else { // swap
        amountOut = IVault(vault).swap(_tokenIn, _tokenOut, _receiver);
    }
    require(amountOut >= _minOut, "Router: insufficient amountOut");
    return amountOut;
}

function sellUSDR(address _token, address _receiver) external override nonReentrant returns (uint256) {
    _validateManager();
    require(whitelistedTokens[_token], "19");
    useSwapPricing = true;
    uint256 usdrAmount = _transferIn(usdr);
    require(usdrAmount > 0, "20");
    updateCumulativeFundingRate(_token);
    uint256 redemptionAmount = getRedemptionAmount(_token, usdrAmount);
    require(redemptionAmount > 0, "21");
    _decreaseUsdrAmount(_token, usdrAmount);
    _decreasePoolAmount(_token, redemptionAmount);
    IUSDR(usdr).burn(address(this), usdrAmount);
    _updateTokenBalance(usdr);
    uint256 feeBasisPoints = getFeeBasisPoints(_token, usdrAmount, mintBurnFeeBasisPoints, taxBasisPoints, false);
    uint256 amountOut = _collectSwapFees(_token, redemptionAmount, feeBasisPoints);
    require(amountOut > 0, "22");
    _transferOut(_token, amountOut, _receiver);
    emit SellUSDR(_receiver, _token, usdrAmount, amountOut, feeBasisPoints);
    useSwapPricing = false;
    return amountOut;
}
```



```
function buyUSDR(address _token, address _receiver) external override nonReentrant returns (uint256) {
    _validateManager();
    require(whitelistedTokens[_token], "16");
    useSwapPricing = true;
    uint256 tokenAmount = _transferIn(_token);
    require(tokenAmount > 0, "17");
    updateCumulativeFundingRate(_token);
    uint256 price = getMinPrice(_token);
    uint256 usdrAmount = tokenAmount.mul(price).div(PRICE_PRECISION);
    usdrAmount = adjustForDecimals(usdrAmount, _token, usdr);
    require(usdrAmount > 0, "18");
    uint256 feeBasisPoints = getFeeBasisPoints(_token, usdrAmount, mintBurnFeeBasisPoints, taxBasisPoints, true);
    uint256 amountAfterFees = _collectSwapFees(_token, tokenAmount, feeBasisPoints);
    uint256 mintAmount = amountAfterFees.mul(price).div(PRICE_PRECISION);
    mintAmount = adjustForDecimals(mintAmount, _token, usdr);
    _increaseUsdrAmount(_token, mintAmount);
    _increasePoolAmount(_token, amountAfterFees);
    IUSDR(usdr).mint(_receiver, mintAmount);
    emit BuyUSDR(_receiver, _token, tokenAmount, mintAmount, feeBasisPoints);
    useSwapPricing = false;
    return mintAmount;
}
```

- **Safety advice**

Add token restrictions to the buyUSDR and sellUSDR functions for buying and selling tokens, prohibiting the use of the same token for the same token.

- **Repair Status**

ROLLUP.FINANCE has Acknowledged.

### 4.3.3 Logic Design Flaw

- Risk Description

In smart contracts, developers design special features for their contracts intended to stabilize the market value of tokens or the life of the project and increase the highlight of the project, however, the more complex the system, the more likely it is to have the possibility of errors. It is in these logic and functions that a minor mistake can lead to serious depasstions from the whole logic and expectations, leaving fatal hidden dangers, such as errors in logic judgment, functional implementation and design and so on.

1. depositFee variable unrestricted maximum

Risk level: Info

The depositFee variable is used as the calculation of fees in the `_collectFees` method. The `BASIS_POINTS_DIVISOR` variable is constant at 10000, but when the depositFee variable is greater than 10000, `BASIS_POINTS_DIVISOR.sub( depositFee)` is calculated as a negative value and a calculation error occurs, since the variable is set by the administrator and its maximum value is not limited.

2. Latest addition of liquidity makes previous proof-of-liquidity tokens cool

Risk level: Info

If a user has added liquidity via `addLiquidity` and `addLiquidityETH`, the previous liquidity proof token is also cooled down when the user adds a new liquidity due to the global variable cooling time of the liquidity funds.

```
function _removeLiquidity(address _account, address _tokenOut,
uint256 _lpAmount, uint256 _minOut, address _receiver) private
returns (uint256) {
    require(_lpAmount > 0, "invalid _lpAmount");
    require(lastAddedAt[_account].add(cooldownDuration) <=
block.timestamp, "cooldown duration not yet passed");
    // calculate aum before sellUSDR
    uint256 aumInUsdr = getAumInUsdr(false);
    uint256 lpSupply = IERC20(lp).totalSupply();
    uint256 usdrAmount = _lpAmount.mul(aumInUsdr).div(lpSupply);
    uint256 usdrBalance = IERC20(usdr).balanceOf(address(this));
    if (usdrAmount > usdrBalance) {
        IUSDR(usdr).mint(address(this), usdrAmount.sub(usdrBalance));
    }
}
```

```
    }  
    IMintable(lp).burn(_account, _lpAmount);  
    IERC20(usdr).transfer(address(vault), usdrAmount);  
    uint256 amountOut = vault.sellUSDR(_tokenOut, _receiver);  
    require(amountOut >= _minOut, "insufficient output");  
    emit RemoveLiquidity(_account, _tokenOut, _lpAmount, aumInUsdr,  
lpSupply, usdrAmount, amountOut);  
    return amountOut;  
}
```

3. The createIncreasePosition method can be called without a fee if minExecutionFee is 0.

Risk level: Info

The createIncreasePosition and createIncreasePositionETH methods check \_executionFee and path when called. When the minExecutionFee variable is zero, all conditions can be bypassed to reach a 0-handle call.

```
function setMinExecutionFee(uint256 _minExecutionFee) external  
onlyAdmin {  
    minExecutionFee = _minExecutionFee;  
    emit SetMinExecutionFee(_minExecutionFee);  
}
```

4. gov may be address(0), suggest adding 0 address judgment.

Risk level: Low

The gov address set by multi-signature is not checked against the new address, and there is a risk that it may be a 0 address.

```
function signalSetGov(address _target, address _gov) external  
override onlyAdmin {  
    bytes32 action = keccak256(abi.encodePacked("setGov", _target,  
_gov));  
    _setPendingAction(action);  
    emit SignalSetGov(_target, _gov, action);  
}
```

- **Safety advice**

1. It is recommended to add a condition to depositFee to prevent the project from running normally if depositFee is greater than 10000.
2. Set a separate cooldown time for each liquidity addition, so that subsequent liquidity additions do not override the previous pledge cooldown time.

- 
3. Add a check to the value at the function that updates the parameter to ensure it is not equal to 0.
  4. Add checksum for 0 address.
    - **Repair Status**
      1. ROLLUP.FINANCE has Acknowledged.
      2. ROLLUP.FINANCE has Acknowledged.
      3. ROLLUP.FINANCE has Acknowledged.
      4. ROLLUP.FINANCE has fixed.

#### 4.3.4 Redundant codes

- Risk description

1. The code overlap between V3 and V4 is too high, the functions are almost the same and there is only one parameter difference.

```
function getVaultTokenInfoV3(address _vault, address
_positionManager, address _weth, uint256 _usdrAmount, address[]
memory _tokens) public view returns (uint256[] memory) {
    uint256 propsLength = 14;
    IVault vault = IVault(_vault);
    IVaultPriceFeed priceFeed = IVaultPriceFeed(vault.priceFeed());
    IBasePositionManager positionManager =
    IBasePositionManager(_positionManager);
    uint256[] memory amounts = new uint256[](_tokens.length *
propsLength);
    for (uint256 i = 0; i < _tokens.length; i++) {
        address token = _tokens[i];
        if (token == address(0)) {
            token = _weth;
        }
        amounts[i * propsLength] = vault.poolAmounts(token);
        amounts[i * propsLength + 1] = vault.reservedAmounts(token);
        amounts[i * propsLength + 2] = vault.usdrAmounts(token);
        amounts[i * propsLength + 3] =
vault.getRedemptionAmount(token, _usdrAmount);
        amounts[i * propsLength + 4] = vault.tokenWeights(token);
        amounts[i * propsLength + 5] = vault.bufferAmounts(token);
        amounts[i * propsLength + 6] = vault.maxUsdrAmounts(token);
        amounts[i * propsLength + 7] = vault.globalShortSizes(token);
        amounts[i * propsLength + 8] =
positionManager.maxGlobalShortSizes(token);
        amounts[i * propsLength + 9] = vault.getMinPrice(token);
```

```
        amounts[i * propsLength + 10] = vault.getMaxPrice(token);
        amounts[i * propsLength + 11] = vault.guaranteedUsd(token);
        amounts[i * propsLength + 12] =
priceFeed.getPrimaryPrice(token, false);
        amounts[i * propsLength + 13] =
priceFeed.getPrimaryPrice(token, true);
    }
    return amounts;
}

function getVaultTokenInfoV4(address _vault, address
_positionManager, address _weth, uint256 _usdrAmount, address[]
memory _tokens) public view returns (uint256[] memory) {
    uint256 propsLength = 15;
    IVault vault = IVault(_vault);
    IVaultPriceFeed priceFeed = IVaultPriceFeed(vault.priceFeed());
    IBasePositionManager positionManager =
IBasePositionManager(_positionManager);
    uint256[] memory amounts = new uint256[](_tokens.length *
propsLength);
    for (uint256 i = 0; i < _tokens.length; i++) {
        address token = _tokens[i];
        if (token == address(0)) {
            token = _weth;
        }
        amounts[i * propsLength] = vault.poolAmounts(token);
        amounts[i * propsLength + 1] = vault.reservedAmounts(token);
        amounts[i * propsLength + 2] = vault.usdrAmounts(token);
        amounts[i * propsLength + 3] =
vault.getRedemptionAmount(token, _usdrAmount);
        amounts[i * propsLength + 4] = vault.tokenWeights(token);
        amounts[i * propsLength + 5] = vault.bufferAmounts(token);
        amounts[i * propsLength + 6] = vault.maxUsdrAmounts(token);
    }
}
```

```
    amounts[i * propsLength + 7] = vault.globalShortSizes(token);
    amounts[i * propsLength + 8] =
positionManager.maxGlobalShortSizes(token);
    amounts[i * propsLength + 9] =
positionManager.maxGlobalLongSizes(token);
    amounts[i * propsLength + 10] = vault.getMinPrice(token);
    amounts[i * propsLength + 11] = vault.getMaxPrice(token);
    amounts[i * propsLength + 12] = vault.guaranteedUsd(token);
    amounts[i * propsLength + 13] =
priceFeed.getPrimaryPrice(token, false);
    amounts[i * propsLength + 14] =
priceFeed.getPrimaryPrice(token, true);
}
return amounts;
}
```

2. There exist functions with different names for exactly the same function, and there may be a waste of deployment gas fees.

```
function claim() external nonReentrant {
    address account = msg.sender;
    IRewardTracker(feeLpTracker).claimForAccount(account, account);
}
```

```
function claimFees() external nonReentrant {
    address account = msg.sender;
    IRewardTracker(feeLpTracker).claimForAccount(account, account);
}
```

3. The `_setupDecimals` method modifies `_decimals`, but the method property is internal and no other method is called.

```
function _setupDecimals(uint8 decimals_) internal {
    _decimals = decimals_;
}
```

- 
- **Safety advice**
    1. You only need to keep a function that gets the most data, to avoid excessive redundant code wasting deployment gas fees.
    2. Remove redundant and useless code.
    3. Remove useless code.
  - **Repair Status**
    1. ROLLUP.FINANCE has Acknowledged.
    2. ROLLUP.FINANCE has Acknowledged.
    3. ROLLUP.FINANCE has fixed.



---

#### 4.3.5 Reentry Attack

- **Risk Description**

The cancelSwapOrder function in the contract will call the \_transferOutETH method to transfer money, but after the transfer, the sendValue method of the caller's address will be called, and the \_transferOutETH method goes to execute \_receiver.sendValue(\_amountOut );, where \_receiver is the address passed in by the user, which can perform other logic or callbacks, \_receiver is a contract address, there is a risk of re-entry, no specific exploitation point has been found yet. Multiple methods will call the \_transferOutETH method to transfer money.

```
function _transferOutETH(uint256 _amountOut, address payable _receiver)
private {
    IWETH(weth).withdraw(_amountOut);
    _receiver.sendValue(_amountOut);
}
```

- **Safety advice**

No available re-entry points have been found yet, but subsequent changes to the contract code would require adding a re-entry prevention mechanism for each external function that calls the function.

- **Audit Results : Passed**

---

#### 4.3.6 Variables are updated

- **Risk description**

When there is a contract logic to obtain rewards or transfer funds, the coder mistakenly updates the value of the variable that sends the funds, so that the user can use the value of the variable that is not updated to obtain funds, thus affecting the normal operation of the project.

- **Audit Results : Passed**

#### 4.3.7 Floating Point and Numeric Precision

- **Risk Description**

In Solidity, the floating-point type is not supported, and the fixed-length floating-point type is not fully supported. The result of the division operation will be rounded off, and if there is a decimal number, the part after the decimal point will be discarded and only the integer part will be taken, for example, dividing 5 pass 2 directly will result in 2. If the result of the operation is less than 1 in the token operation, for example, 4.9 tokens will be approximately equal to 4, bringing a certain degree of The tokens are not only the tokens of the same size, but also the tokens of the same size. Due to the economic properties of tokens, the loss of precision is equivalent to the loss of assets, so this is a cumulative problem in tokens that are frequently traded.

- **Audit Results : Passed**

---

#### 4.3.8 Default Visibility

- **Risk description**

In Solidity, the visibility of contract functions is public pass default. therefore, functions that do not specify any visibility can be called externally pass the user. This can lead to serious vulnerabilities when developers incorrectly ignore visibility specifiers for functions that should be private, or visibility specifiers that can only be called from within the contract itself. One of the first hacks on Parity's multi-signature wallet was the failure to set the visibility of a function, which defaults to public, leading to the theft of a large amount of money.

- **Audit Results : Passed**

#### 4.3.9 tx.origin authentication

- **Risk Description**

tx.origin is a global variable in Solidity that traverses the entire call stack and returns the address of the account that originally sent the call (or transaction). Using this variable for authentication in a smart contract can make the contract vulnerable to phishing-like attacks.

- **Audit Results : Passed**

---

#### 4.3.10 Faulty constructor

- **Risk description**

Prior to version 0.4.22 in solidity smart contracts, all contracts and constructors had the same name. When writing a contract, if the constructor name and the contract name are not the same, the contract will add a default constructor and the constructor you set up will be treated as a normal function, resulting in your original contract settings not being executed as expected, which can lead to terrible consequences, especially if the constructor is performing a privileged operation.

- **Audit Results : Passed**

#### 4.3.11 Unverified return value

- **Risk description**

Three methods exist in Solidity for sending tokens to an address: `transfer()`, `send()`, `call.value()`. The difference between them is that the transfer function throws an exception throw when sending fails, rolls back the transaction state, and costs 2300gas; the send function returns false when sending fails and costs 2300gas; the call.value method returns false when sending fails and costs all gas to call, which will lead to the risk of reentrant attacks. If the send or call.value method is used in the contract code to send tokens without checking the return value of the method, if an error occurs, the contract will continue to execute the code later, which will lead to the thought result.

- **Audit Results : Passed**

#### 4.3.12 Insecure random numbers

- **Risk Description**

All transactions on the blockchain are deterministic state transition operations with no uncertainty, which ultimately means that there is no source of entropy or randomness within the blockchain ecosystem. Therefore, there is no random number function like `rand()` in Solidity. Many developers use future block variables such as block hashes, timestamps, block highs and lows or Gas caps to generate random numbers. These quantities are controlled pass the miners who mine them and are therefore not truly random, so using past or present block variables to generate random numbers could lead to a destructive vulnerability.

- **Audit Results : Passed**

#### 4.3.13 Timestamp Dependency

- **Risk description**

In blockchains, data block timestamps (`block.timestamp`) are used in a variety of applications, such as functions for random numbers, locking funds for a period of time, and conditional statements for various time-related state changes. Miners have the ability to adjust the timestamp as needed, for example `block.timestamp` or the alias `now` can be manipulated pass the miner. This can lead to serious vulnerabilities if the wrong block timestamp is used in a smart contract. This may not be necessary if the contract is not particularly concerned with miner manipulation of block timestamps, but care should be taken when developing the contract.

- **Audit Results : Passed**

#### 4.3.14 Transaction order dependency

- **Risk description**

In a blockchain, the miner chooses which transactions from that pool will be included in the block, which is usually determined pass the gasPrice transaction, and the miner will choose the transaction with the highest transaction fee to pack into the block. Since the information about the transactions in the block is publicly available, an attacker can watch the transaction pool for transactions that may contain problematic solutions, modify or revoke the attacker's privileges or change the state of the contract to the attacker's detriment. The attacker can then take data from this transaction and create a higher-level transaction gasPrice and include its transactions in a block before the original, which will preempt the original transaction solution.

- **Audit Results : Passed**

#### 4.3.15 Delegatecall

- **Risk Description**

In Solidity, the delegatecall function is the standard message call method, but the code in the target address runs in the context of the calling contract, i.e., keeping msg.sender and msg.value unchanged. This feature supports implementation libraries, where developers can create reusable code for future contracts. The code in the library itself can be secure and bug-free, but when run in another application's environment, new vulnerabilities may arise, so using the delegatecall function may lead to unexpected code execution.

- **Audit Results : Passed**

---

#### 4.3.16 Call

- **Risk Description**

The call function is similar to the delegatecall function in that it is an underlying function provided pass Solidity, a smart contract writing language, to interact with external contracts or libraries, but when the call function method is used to handle an external Standard Message Call to a contract, the code runs in the environment of the external contract/function The call function is used to interact with an external contract or library. The use of such functions requires a determination of the security of the call parameters, and caution is recommended. An attacker could easily borrow the identity of the current contract to perform other malicious operations, leading to serious vulnerabilities.

- **Audit Results : Passed**

#### 4.3.17 Denial of Service

- **Risk Description**

Denial of service attacks have a broad category of causes and are designed to keep the user from making the contract work properly for a period of time or permanently in certain situations, including malicious behavior while acting as the recipient of a transaction, artificially increasing the gas required to compute a function causing gas exhaustion (such as controlling the size of variables in a for loop), misuse of access control to access the private component of the contract, in which the Owners with privileges are modified, progress state based on external calls, use of obfuscation and oversight, etc. can lead to denial of service attacks.

- **Audit Results : Passed**

#### 4.3.18 Fake recharge vulnerability

- **Risk Description**

The success or failure (true or false) status of a token transaction depends on whether an exception is thrown during the execution of the transaction (e.g., using mechanisms such as `require/assert/revert/throw`). When a user calls the transfer function of a token contract to transfer funds, if the transfer function runs normally without throwing an exception, the transaction will be successful or not, and the status of the transaction will be true. When `balances[msg.sender] < _value` goes to the else logic and returns false, no exception is thrown, but the transaction acknowledgement is successful, then we believe that a mild if/else judgment is an undisciplined way of coding in sensitive function scenarios like transfer, which will lead to Fake top-up vulnerability in centralized exchanges, centralized wallets, and token contracts.

- **Audit Results : Passed**

#### 4.3.19 Short Address Attack Vulnerability

- **Risk Description**

In Solidity smart contracts, when passing parameters to a smart contract, the parameters are encoded according to the ABI specification. the EVM runs the attacker to send encoded parameters that are shorter than the expected parameter length. For example, when transferring money on an exchange or wallet, you need to send the transfer address address and the transfer amount value. The attacker could send a 19-paste address instead of the standard 20-paste address, in which case the EVM would fill in the 0 at the end of the encoded parameter to make up the expected length, which would result in an overflow of the final transfer amount parameter value, thus changing the original transfer amount.

- **Audit Results : Passed**



---

#### 4.3.20 Uninitialized storage pointer

- **Risk description**

EVM uses both storage and memory to store variables. Local variables within functions are stored in storage or memory pass default, depending on their type. uninitialized local storage variables could point to other unexpected storage variables in the contract, leading to intentional or unintentional vulnerabilities.

- **Audit Results : Passed**

#### 4.3.21 Frozen Account bypass

- **Risk Description**

In the transfer operation code in the contract, detect the risk that the logical functionality to check the freeze status of the transfer account exists in the contract code and can be passpassed if the transfer account has been frozen.

- **Audit Results : Passed**

#### 4.3.22 Uninitialized

- **Risk description**

The initialize function in the contract can be called pass another attacker before the owner, thus initializing the administrator address.

- **Audit Results : Passed**

---

#### 4.3.23 Integer Overflow

- **Risk Description**

Integer overflows are generally classified as overflows and underflows. The types of integer overflows that occur in smart contracts include three types: multiplicative overflows, additive overflows, and subtractive overflows. In Solidity language, variables support integer types in steps of 8, from uint8 to uint256, and int8 to int256, integers specify fixed size data types and are unsigned, for example, a uint8 type, can only be stored in the range 0 to  $2^8-1$ , that is, [0,255] numbers, a uint256 type can only store numbers in the range 0 to  $2^{256}-1$ . This means that an integer variable can only have a certain range of numbers represented, and cannot exceed this formulated range. Exceeding the range of values expressed pass the variable type will result in an integer overflow vulnerability.

- **Audit Results : Passed**

---

## 5. Security Audit Tool

Tool name	Tool Features
Oyente	Can be used to detect common bugs in smart contracts
securify	Common types of smart contracts that can be verified
MAIAN	Multiple smart contract vulnerabilities can be found and classified
Lunaray Toolkit	self-developed toolkit

---

## Disclaimer:

Lunaray Technology only issues a report and assumes corresponding responsibilities for the facts that occurred or existed before the issuance of this report, Since the facts that occurred after the issuance of the report cannot determine the security status of the smart contract, it is not responsible for this.

Lunaray Technology conducts security audits on the security audit items in the project agreement, and is not responsible for the project background and other circumstances, The subsequent on-chain deployment and operation methods of the project party are beyond the scope of this audit.

This report only conducts a security audit based on the information provided by the information provider to Lunaray at the time the report is issued, If the information of this project is concealed or the situation reflected is inconsistent with the actual situation, Lunaray Technology shall not be liable for any losses and adverse effects caused thereby.

There are risks in the market, and investment needs to be cautious. This report only conducts security audits and results announcements on smart contract codes, and does not make investment recommendations and basis.



<https://lunaray.co>



<https://github.com/lunaraySec>



[https://twitter.com/lunaray\\_Sec](https://twitter.com/lunaray_Sec)



<http://t.me/lunaraySec>