LUNARAY
BLOCKCHAINSECURITY

SMART CONTRACT SECURITY AUDIT REPORT

For Roguex

14 December 2023

lunaray.co

Table of Contents

1. Overview

On Nov 20, 2023, the security team of Lunaray Technology received the security audit request of the **ROGUEX project**. The team completed the audit of the **ROGUEX smart contract** on Dec 14, 2023. During the audit process, the security audit experts of Lunaray Technology and the ROGUEX 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 communicat and feedback with ROGUEX 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 ROGUEX smart contract security audit: **Passed**

Audit Report Hash:

21CB3C244E6EDEDED5CB4DA02A985C3AC98A2488001F630F18D0F23948BEB55D

2. Background

2.1 Project Description

Project name	RogueX
Contract type	Spot and perpetual trading
Code language	Solidity
Public chain	Arbitrum
Project website	https://roguex.io
Introduction	RogueX innovates with an AMM that merges perpetual trading into liquidity pools, enhancing asset utilization and supporting leveraged trades for a diverse range of tokens, especially memecoins.
Contract file	HypervisorFactory.sol MasterChefFactory.sol Minter.sol NonfungiblePositionManager.sol PerpOrderbook.sol PerpRouter.sol RewardsDistributor.sol RoguexFactory.sol RoxPerpPool.sol RoxPosnPool.sol RoxSpotPool.sol ROXToken.sol RoxUtils.sol SwapMinningFactory.sol SwapRouter.sol VoterV2.sol VotingEscrow.sol VotingRewardV2Factory.sol

2.2 Audit Range

Smart contract file name and corresponding SHA256：

Name	SHA256
HypervisorFactory.sol	A9BFE4D14A36102C91FAA91EDAC1E3F2781CB254A9453769AEE88583908E4F70
MasterChefFactory.sol	7AE10C4770754082739BA0B8E2C426ACA3DB7C41734A03E2B50D37222C8711EF
Minter.sol	DDCBFAAA07D3E043035BFC1D487408CF636DD4BA718D29D33F29D9BC27688DF1
NonfungiblePositionManager.sol	34B0EB98704ABA84C9DC78B1DDB367E1E44938B9180FCB7BBD351ED74E803C8B
PerpOrderbook.sol	2E54E744422F87638AB885365D54A2F0B1A6C60F5B9424DF983BBC98622A5ADF
PerpRouter.sol	2718820DD529ADFC6066493EBEF0F3ACB98D4DFD2A73361CFD6410FABAC90BE0
RewardsDistributor.sol	B1995F4DE73179F9C75456B5D6DA9259712B2DF7B63591DBB220BBF354159A44
RoguexFactory.sol	4E9F2AE160069A03F06BE0FB8D506B73D7632DE5AE2F3D55438D7BFDBF27DF12
RoxPerpPool.sol	075BE40F82964A2618A2095551FE9A5F6285B7B7EFA2411A5DF3CE59EB8422EF
RoxPosnPool.sol	ADDC941F035DD103C3921680A53B1FCE2631B203C1C35DD89B43B7441472B4F5
RoxSpotPool.sol	A2863E89CFE131DC9563B5C4DE16673962440E9473A0A892C7E29AFD4A3ACDB1
ROXToken.sol	77B26EA8F6E2CAD8F7B085AC061757598FBFA74446518E42F1043B9B319B9474
RoxUtils.sol	AC0A018B21568A02701AA982EF8C3C8735A110B94EE76C611C3CD33859907F61
SwapMinningFactory.sol	7B85EEA6EB1C2B8FD9737260238D6B3860074EFF13EFF40786256047F5F5217F
SwapRouter.sol	B2662D936569100C3ED1F733FF55594FE7E6525D8A671F9B2DE90D7AB7F4353C

VoterV2.sol	5AF0565B00C1A2BFD0BF0833841D0105A8B684F7269C02C023D94751AF821DAF
VotingEscrow.sol	E554E8D5AABFEDAD2CD4A5143D78A2ADFEF98A89556A347FC6D9522945A1EAAF
VotingRewardV2Factory.sol	D9FEFE42900FF1AEB2D8CDBFEB9B93DDAA95337868654D3070AADABF49330E26

3. Project contract details

3.1 Contract Overview

VaultPriceFeedV2 Contract

The contract is primarily used to obtain price data for different tokens. The contract provides multiple price sources, including Pyth, AMM and carefully maintained price sources by the project, which can be enabled or disabled based on configuration parameters set by the administrator.

FastPriceFeedV2 Contract

The main implemented functionality of the contract is to provide token price data and to allow adding extended information such as cumulative price change data. It also allows managing update policies for price data, including time intervals and delays, as well as configuring extension parameters for price updates. In addition, the contract supports enabling/disabling price data extensions such as price variance and price monitoring, and allows administrators to manage signers, updaters, and other associated contracts. Specific functions include price data management, price extension information logging, price calculation, price update method setting, configuration parameter provisioning and administrative privilege control.

3.2 Contract details

RoxPerpPool Contract

Name	Parameter	Attributes
priceSlot	uint psId	external
rgFeeSlot	none	external
prInfo	uint256 timePr	external
updateSwapFee	int24 tick bool zeroForOne uint256 feeToken uint256 liquidity	external
encodeSlots	uint256 prStart uint256 prEnd bool isPrice	public
increasePosition	address _account uint256 _sizeDelta bool _long0	external
_validSender	address _owner	private
decreasePosition	bytes32 _key uint256 _collateralDelta uint256 _sizeDelta address _feeRecipient bool _toETH	external
_increaseReserve	uint256 _delta bool _token0	private
_decreaseReserve	uint256 _delta bool _token0	private
_delPosition	bytes32 _key	private
_transferIn	bool _isToken0	private
_transferOut0	uint256 _amount0 address _recipient bool _toETH	private
_transferOut1	uint256 _amount1 address _recipient	private

	bool _toETH	
balance0	none	private
balance1	none	private
getPositionByKey	bytes32 _key	public
settle	address _recipient bool _is0 bool _burn uint256 _tokenAmount uint256 _feeAmount	internal
tPid	bool l0	public
updateFundingRate	none	public

RoxUtils Contract

Name	Parameter	Attributes
setTime	uint32 _time	external
setFactor	uint256 _kMax uint256 _powF uint32 _countMin	external
getSqrtTwapX96	address spotPool	public
getSqrtTwapX96Sec	address spotPool uint32 secAgo	public
nextInitializedTickWithinOneWord	address spotPool int24 tick bool lte int24 tickSpacing	public
estimate	address spotPool bool zeroForOne int256 amountSpecified	public
getLiquidityArraySpecifiedStart	address spotPool int24 curTick int24 tickStart	public

	bool isToken0 uint256 amount	
calLiqSpecifiedAmount0	address spotPool TradeData.LiqCalState state TradeData.PriceRangeLiq prState	private
calLiqSpecifiedAmount1	address spotPool TradeData.LiqCalState state TradeData.PriceRangeLiq prState	private
token0t1NoSpl	address _spotPool uint256 _amount0	public
token1t0NoSpl	address _spotPool uint256 _amount1	public
gOpenPrice	address _roguPool uint256 _sizeDelta bool _long0 bool _isSizeCor	public
getClosePrice	address _perpPool bool _long0 uint256 _sizeDelta bool _isCor	public
gClosePrice	address _roguPool uint256 _sizeDelta TradeData.TradePosition tP bool _isCor	public
countClose	address _perpPool bool long0 uint32 minC	public
_factor	address _spotPool TradeData.TradePosition tP uint256 _sizeDelta uint256 _sqrtSpd	private
validPosition	uint256 collateral uint256 size	public
collectPosFee	uint256 size	public

ROXToken Contract

Name	Parameter	Attributes
setMinter	address _minter	external
initialMint	address _recipient	external
approve	address _spender uint _value	external
_mint	address _to uint _amount	internal
_transfer	address _from address _to uint _value	internal
transfer	address _to uint _value	external
transferFrom	address _from address _to uint _value	external
burn	uint256 amount	external
mint	address account uint amount	external

PerpRouter Contract

Name	Parameter	Attributes
setSwapMining	address addr	public
setUtils	address _roguUtils address _perpOrderbook	external
increasePositionOrder	address _account address _perpPool uint256 _tokenAmount uint256 _sizeDelta bool _long0 uint160 _sqrtPriceX96	external

_increasePosition	address _account address _perpPool uint256 _sizeDelta bool _long0	private
liquidatePosition	address _perpPool bytes32 _key	external
decreasePosition	address _account address _perpPool uint256 _collateralDelta uint256 _sizeDelta bool _long0 bool _toETH uint160 _sqrtPriceX96	external
execTakingProfitSet	address _perpPool bytes32 _posKey	external
getPositionKeys	address _account	public
_sender	none	private

Minter Contract

Name	Parameter	Attributes
max	uint a uint b	internal
min	uint a uint b	internal
sqrt	uint y	internal
cbrt	uint256 n	internal
checkpoint_token	none	external
checkpoint_total_supply	none	external
totalSupply	none	external
approve	address spender uint value	external
_ve	none	external
notifyRewardAmount	uint amount	external
token	none	external

totalSupply	none	external
initialize	none	external
calculate_emission	none	public
weekly_emission	none	public
circulating_emission	none	public
calculate_growth	uint _minted	public
update_period	none	external

PerpOrderbook Contract

Name	Parameter	Attributes
setUtils	address _roguUtils address _tradeRouter	external
getIncreaseOrder	uint256 _id	public
getPendingIncreaseOrdersKeys	address _account	public
getPendingIncreaseOrders	address _account	public
pendingIncreaseOrdersNum	address _account	public
isIncreaseOrderKeyAlive	uint256 _increaseKey	public
getDecreaseOrderByKey	uint256 _decreaseKey	public
getPendingDecreaseOrdersKeys	address _account	public
getPendingDecreaseOrders	address _account	public
pendingDecreaseOrdersNum	address _account	public
isDecreaseOrderKeyAlive	uint256 _decreaseKey	public
getPendingOrders	address _account	public
createIncreaseOrder	address _perpPool uint128 _tokenAmount uint128 _exeFee	external

	uint256 _sizeDelta uint160 _triggerPriceSqrtX96 bool _long0 bool _triggerAboveThreshold bool _shouldWarp	
executeIncreaseOrder	uint256 _key address _feeReceipt	external
cancelIncreaseOrder	uint256 _orderIndex	public
updateIncreaseOrder	uint256 _key int256 _sizeDelta int256 _colDelta int256 _feeDelta uint160 _triggerPrice bool _triggerAboveThreshold	public
createDecreaseOrder	address _perpPool uint128 _exeFee uint128 _colDelta uint256 _sizeDelta uint160 _triggerPriceSqrtX96 bool _long0 bool _triggerAboveThreshold bool _shouldWarp	external
cancelDecreaseOrder	uint256 _orderIndex	public
executeDecreaseOrder	uint256 _key address _feeReceipt	external
updateDecreaseOrder	uint256 _key int256 _sizeDelta int256 _colDelta int256 _feeDelta uint160 _triggerPrice bool _triggerAboveThreshold	public
_transferInETH	none	private
_transferOutETH	uint256 _amountOut address payable _receiver	private
validatePositionOrderPrice	address _perpPool bool _triggerAboveThreshold uint256 _triggerPrice	public

RoxPosnPool Contract

Name	Parameter	Attributes
positions	bytes32 key	public
positionsSum	bytes32 key	public
increaseLiquidity	address owner int24 tickLower int24 tickUpper uint128 liquidityDelta	onlySpotPool
collect	bytes32 _key uint128 _amount0Requested uint128 _amount1Requested	onlySpotPool
updateFee	bytes32 _key	public
decreaseLiquidity	bytes32 _key uint128 liquidityDelta int24 tick uint160 sqrtPriceX96	onlySpotPool
estimateDecreaseLiquidity	bytes32 _key uint128 liquidityDelta int24 tick uint160 sqrtPriceX96	external
pendingFee	bytes32 _key	external

VoterV2 Contract

Name	Parameter	Attributes
manager	none	external
renounceManagement	none	external
pushManagement	address newOwner_	external
pullManagement	none	external
manager	none	public
renounceManagement	none	onlyManager
pushManagement	address newOwner_	onlyManager

pullManagement	none	public
max	uint a uint b	internal
min	uint a uint b	internal
sqrt	uint y	internal
cbrt	uint256 n	internal
totalSupply	none	external
transfer	address recipient uint amount	external
decimals	none	external
symbol	none	external
transferFrom	address sender address recipient uint amount	external
allowance	address owner address spender	external
approve	address spender uint value	external
getReward	address _account uint round	external
deposit	uint256 _amount address _recipient	external
notifyRewardAmount	address token uint amount	external
active_period	none	external
update_period	none	external
token	none	external
team	none	external
voting	uint tokenId	external
abstain	uint tokenId	external
totalSupply	none	external
token0	none	external
token1	none	external
owner	none	external
getPool	address tokenA	external

	address tokenB uint24 fee	
notifyRewardAmount	address token uint256 _amount	external
getReward	address _account uint round	external
deposit	uint256 _amount address _recipient	external
notifyRewardAmount	address token uint256 _amount	external
getReward	address _account	external
setMinter	address _minter	onlyManager
reset	uint _tokenId	onlyNewEpoch
resetNoVoted	uint _tokenId	external
_reset	uint _tokenId	internal
poke	uint _tokenId	external
notifyFeeAmount	address poolAddrss uint amount0 uint amount1	external
createGauge	address _pool address _hypervisor	external
killGauge	address _gauge	onlyManager
reviveGauge	address _gauge	onlyManager
_epochTimestamp	none	public
weights	address _pool	public
weightsAt	address _pool uint256 _time	public
totalWeight	none	public
totalWeightAt	uint256 _time	public
getPools	none	external
depositSwap	address _pool uint256 _amount address _recipient	external
notifyRewardAmount	uint amount	external

_updateFor	address _gauge	private
_updateForFee	address _gauge	internal
_distribute	address _gauge	internal
_safeTransferFrom	address token address from address to uint256 value	internal

SwapRouter Contract

Name	Parameter	Attributes
setSwapMining	address addr	onlyOwner
getPool	address tokenA address tokenB uint24 fee	private
swapCallback	int256 amount0Delta int256 amount1Delta bytes _data	external
exactInputInternal	uint256 amountIn address recipient uint160 sqrtPriceLimitX96 SwapCallbackData data	private
exactInputSingle	ExactInputSingleParams params	external
exactInput	ExactInputParams params	external
exactOutputInternal	uint256 amountOut address recipient uint160 sqrtPriceLimitX96 SwapCallbackData data	private
exactOutputSingle	ExactOutputSingleParams params	external
exactOutput	ExactOutputParams params	external

RewardsDistributor Contract

Name	Parameter	Attributes
max	uint a uint b	internal
min	uint a uint b	internal
sqrt	uint y	internal
cbrt	uint256 n	internal
totalSupply	none	external
transfer	address recipient uint amount	external
decimals	none	external
symbol	none	external
transferFrom	address sender address recipient uint amount	external
allowance	address owner address spender	external
approve	address spender uint value	external
token	none	external
team	none	external
epoch	none	external
point_history	uint loc	external
user_point_history	uint tokenId uint loc	external
user_point_epoch	uint tokenId	external
voting	uint tokenId	external
abstain	uint tokenId	external
attach	uint tokenId	external
detach	uint tokenId	external
checkpoint	none	external
deposit_for	uint tokenId	external

	uint value	
totalSupply	none	external
timestamp	none	external
_checkpoint_token	none	internal
checkpoint_token	none	external
_find_timestamp_epoch	address ve uint _timestamp	internal
_find_timestamp_user_epoch	address ve uint tokenId uint _timestamp uint max_user_epoch	internal
ve_for_at	uint _tokenId uint _timestamp	external
_checkpoint_total_supply	none	internal
checkpoint_total_supply	none	external
_claim	uint _tokenId address ve uint _last_token_time	internal
_claimable	uint _tokenId address ve uint _last_token_time	internal
claimable	uint _tokenId	external
claim	uint _tokenId	external
setDepositor	address _depositor	external

HypervisorFactory Contract

Name	Parameter	Attributes
positionsSum	bytes32 key	external
mulDiv	uint256 a uint256 b uint256 denominator	internal
mulDivRoundingUp	uint256 a	internal

	uint256 b uint256 denominator	
divRoundingUp	uint256 x uint256 y	internal
getAmount0Delta	uint160 sqrtRatioAX96 uint160 sqrtRatioBX96 uint128 liquidity bool roundUp	internal
getAmount1Delta	uint160 sqrtRatioAX96 uint160 sqrtRatioBX96 uint128 liquidity bool roundUp	internal
_msgSender	none	internal
_msgData	none	internal
owner	none	public
renounceOwnership	none	onlyOwner
transferOwnership	address newOwner	onlyOwner
recover	bytes32 hash uint8 v bytes32 r bytes32 s	internal
toEthSignedMessageHash	bytes32 hash	internal
_domainSeparatorV4	none	internal
_buildDomainSeparator	bytes32 typeHash bytes32 name bytes32 version	private
_hashTypedDataV4	bytes32 structHash	internal
_getChainId	none	private
permit	address owner address spender uint256 value uint256 deadline uint8 v bytes32 r bytes32 s	external
nonces	address owner	external

DOMAIN_SEPARATOR	none	external
totalSupply	none	external
balanceOf	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
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
_setupDecimals	uint8 decimals_	internal
_beforeTokenTransfer	address from address to uint256 amount	internal
current	Counter counter	internal
increment	Counter counter	internal
decrement	Counter counter	internal
permit	address owner address spender uint256 value uint256 deadline uint8 v bytes32 r bytes32 s	public
nonces	address owner	public
DOMAIN_SEPARATOR	none	external
isContract	address account	internal
sendValue	address payable recipient uint256 amount	internal
functionCall	address target bytes data string errorMessage	internal
functionCallWithValue	address target bytes data uint256 value string errorMessage	internal
functionStaticCall	address target bytes data string errorMessage	internal
functionDelegateCall	address target bytes data	internal

	string errorMessage	
_verifyCallResult	bool success bytes returndata string errorMessage	private
safeTransfer	IERC20 token address to uint256 value	internal
safeTransferFrom	IERC20 token address from address to uint256 value	internal
safeApprove	IERC20 token address spender uint256 value	internal
safeIncreaseAllowance	IERC20 token address spender uint256 value	internal
safeDecreaseAllowance	IERC20 token address spender uint256 value	internal
_callOptionalReturn	IERC20 token bytes data	private
collect	address recipient int24 tickLower int24 tickUpper uint128 amount0Requested uint128 amount1Requested	external
burn	int24 tickLower int24 tickUpper uint128 amount	external
mint	address recipient int24 tickLower int24 tickUpper uint128 amount bytes data	external
token0	none	external

roxPosnPool	none	external
token1	none	external
tickSpacing	none	external
slot0	none	external
positions	bytes32 key	external
liquidity	none	external
toUint128	uint256 x	private
getLiquidityForAmount0	uint160 sqrtRatioAX96 uint160 sqrtRatioBX96 uint256 amount0	internal
getLiquidityForAmount1	uint160 sqrtRatioAX96 uint160 sqrtRatioBX96 uint256 amount1	internal
getLiquidityForAmounts	uint160 sqrtRatioX96 uint160 sqrtRatioAX96 uint160 sqrtRatioBX96 uint256 amount0 uint256 amount1	internal
getAmount0ForLiquidity	uint160 sqrtRatioAX96 uint160 sqrtRatioBX96 uint128 liquidity	internal
getAmount1ForLiquidity	uint160 sqrtRatioAX96 uint160 sqrtRatioBX96 uint128 liquidity	internal
getAmountsForLiquidity	uint160 sqrtRatioX96 uint160 sqrtRatioAX96 uint160 sqrtRatioBX96 uint128 liquidity	internal
max	uint256 a uint256 b	internal
min	uint256 a uint256 b	internal
sqrt	uint y	internal
average	uint256 a uint256 b	internal

mul	int256 a int256 b	internal
div	int256 a int256 b	internal
sub	int256 a int256 b	internal
add	int256 a int256 b	internal
mintCallback	uint256 amount0Owed uint256 amount1Owed bytes data	external
getSqrtRatioAtTick	int24 tick	internal
getTickAtSqrtRatio	uint160 sqrtPriceX96	internal
notifyFeeAmount	address poolAddrss uint amount0 uint amount1	external
createGauge	address _pool address _hypervisor	external
deposit	uint256 _amount address _recipient	external
withdraw	uint256 _amount address _recipient	external
getPriceByTick	int24 _tick	internal
getTikcUpperAndLower	uint256 price	public
deposit	uint256 deposit0 uint256 deposit1 address to address from	external
adjustRange	none	public
_zeroBurn	int24 tickLower int24 tickUpper	internal
zeroBurn	none	internal
pullLiquidity	int24 tickLower int24 tickUpper uint128 shares	onlyOwner

withdraw	uint256 shares address to address from	external
rebalance	none	onlyOwner
_rebalance	none	internal
_compound	none	internal
compound	none	onlyOwner
addLiquidity	int24 tickLower int24 tickUpper uint256 amount0 uint256 amount1	onlyOwner
_beforeTokenTransfer	address from address to uint256 amount	internal
_mintLiquidity	int24 tickLower int24 tickUpper uint128 liquidity address payer	internal
_burnLiquidity	int24 tickLower int24 tickUpper uint128 liquidity address to bool collectAll	internal
_liquidityForShares	int24 tickLower int24 tickUpper uint256 shares	internal
_position	int24 tickLower int24 tickUpper	internal
mintCallback	uint256 amount0 uint256 amount1 bytes data	external
getTotalAmounts	none	public
getUserAmounts	address account	public
getBasePosition	none	public
getLimitPosition	none	public
_amountsForLiquidity	int24 tickLower	internal

	int24 tickUpper uint128 liquidity	
_liquidityForAmounts	int24 tickLower int24 tickUpper uint256 amount0 uint256 amount1	internal
currentTick	none	public
getDepositAmountRatio	none	public
currentPrice	none	public
_uint128Safe	uint256 x	internal
transferOwnership	address newOwner	onlyOwner
feeAmountTickSpacing	uint24 fee	external
getPool	address tokenA address tokenB uint24 fee	external
enableFeeAmount	uint24 fee int24 tickSpacing	external
allHypervisorsLength	none	external
createHypervisor	address tokenA address tokenB uint24 fee string name string symbol	external

MasterChefFactory Contract

Name	Parameter	Attributes
manager	none	external
renounceManagement	none	external
pushManagement	address newOwner_	external
pullManagement	none	external
manager	none	public
renounceManagement	none	onlyManager
pushManagement	address newOwner_	onlyManager

pullManagement	none	public
max	uint a uint b	internal
min	uint a uint b	internal
sqrt	uint y	internal
cbrt	uint256 n	internal
totalSupply	none	external
transfer	address recipient uint amount	external
decimals	none	external
symbol	none	external
transferFrom	address sender address recipient uint amount	external
allowance	address owner address spender	external
approve	address spender uint value	external
distribute	address _gauge	external
rewardPerToken	address token	public
lastTimeRewardApplicabl e	address token	public
getRewardList	none	public
getReward	address _account	external
_getReward	address _account	internal
earned	address token address _account	public
deposit	uint256 _amount address _recipient	external
withdraw	uint256 _amount address _recipient	external
_updateRewards	address _account	internal
notifyRewardAmount	address token uint256 _amount	external
epochNext	uint256 timestamp	internal
setVoter	address _voter	onlyManager

SwapMinningFactory Contract

Name	Parameter	Attributes
manager	none	external
renounceManagement	none	external
pushManagement	address newOwner_	external
pullManagement	none	external
manager	none	public
renounceManagement	none	onlyManager
pushManagement	address newOwner_	onlyManager
pullManagement	none	public
max	uint a uint b	internal
min	uint a uint b	internal
sqrt	uint y	internal
cbrt	uint256 n	internal
totalSupply	none	external
transfer	address recipient uint amount	external
decimals	none	external
symbol	none	external
transferFrom	address sender address recipient uint amount	external
allowance	address owner address spender	external
approve	address spender uint value	external
distribute	address _gauge	external
rewardPerToken	address token uint round	public
lastTimeRewardApplicabl e	address token uint round	public

getRewardList	none	public
getReward	address _account uint round	external
earned	address token address _account uint round	public
deposit	uint256 _amount address _recipient	external
_deposit	uint256 _amount address _recipient	internal
_updateRewards	address _account uint round	internal
notifyRewardAmount	address token uint256 _amount	external
getCurrRound	none	public
epochNext	uint256 timestamp	internal
setVoter	address _voter	onlyManager

NonfungiblePositionManager Contract

Name	Parameter	Attributes
positions	uint256 tokenId	external
cachePoolKey	address pool PoolAddress.PoolKey poolKey	private
mint	MintParams params	external
tokenURI	uint256 tokenId	public
baseURI	none	public
increaseLiquidity	IncreaseLiquidityParams params	external
decreaseLiquidity	DecreaseLiquidityParams params	external
collect	CollectParams params	external
burn	uint256 tokenId	external
_getAndIncrementNonce	uint256 tokenId	internal

RoxSpotPool Contract

Name	Parameter	Attributes
_blockTimestamp	none	internal
balance0	none	private
balance1	none	private
increaseObservationCardinalityNext	uint16 observationCardinalityNext	external
initialize	uint160 sqrtPriceX96	external
_modifyPosition	PoolData.ModifyPositionParams params	private
_updateLiquidity	int24 tickLower int24 tickUpper int128 liquidityDelta int24 tick uint32 time	private
mint	address recipient int24 tickLower int24 tickUpper uint128 amount bytes data	external
collect	address recipient int24 tickLower int24 tickUpper uint128 amount0Requested uint128 amount1Requested	external
collectN	address recipient int24 tickLower int24 tickUpper uint128 amount0Requested uint128 amount1Requested	onlyNftManager
_collect	address owner address recipient int24 tickLower int24 tickUpper uint128 amount0Requested	private

	uint128 amount1Requested	
burn	int24 tickLower int24 tickUpper uint128 amount	external
burnN	address owner int24 tickLower int24 tickUpper uint128 amount	onlyNftManager
_burn	address owner int24 tickLower int24 tickUpper uint128 amount	private
updatePnl	int24 tickLower int24 tickUpper int24 slot0tick int128 liquidityDelta	onlyPerpPool
perpSettle	uint256 amount bool is0 bool isBurn address recipient	onlyPerpPool
swap	address recipient bool zeroForOne int256 amountSpecified uint160 sqrtPriceLimitX96 bytes data	external
availableReserve	bool _l0 bool _l1	public

VotingRewardFactoryV2Contract

Name	Parameter	Attributes
manager	none	external
renounceManagement	none	external
pushManagement	address newOwner_	external
pullManagement	none	external
manager	none	public

renounceManagement	none	onlyManager
pushManagement	address newOwner_	onlyManager
pullManagement	none	public
max	uint a uint b	internal
min	uint a uint b	internal
sqrt	uint y	internal
cbrt	uint256 n	internal
totalSupply	none	external
transfer	address recipient uint amount	external
decimals	none	external
symbol	none	external
transferFrom	address sender address recipient uint amount	external
allowance	address owner address spender	external
approve	address spender uint value	external
distribute	address _gauge	external
rewardPerToken	address token uint round	public
lastTimeRewardApplicabl e	address token uint round	public
getRewardList	none	public
getReward	address _account uint round	external
earned	address token address _account uint round	public
deposit	uint256 _amount address _recipient	external
_deposit	uint256 _amount	internal

	address _recipient	
_updateRewards	address _account uint round	internal
notifyRewardAmount	address token uint256 _amount	external
getCurrRound	none	public
epochNext	uint256 timestamp	internal
setVoter	address _voter	onlyManager

VotingEscrow Contract

Name	Parameter	Attributes
supportsInterface	bytes4 interfaceId	external
balanceOf	address owner	external
ownerOf	uint256 tokenId	external
safeTransferFrom	address from address to uint256 tokenId	external
transferFrom	address from address to uint256 tokenId	external
approve	address to uint256 tokenId	external
setApprovalForAll	address operator bool approved	external
getApproved	uint256 tokenId	external
isApprovedForAll	address owner address operator	external
name	none	external
symbol	none	external
tokenURI	uint256 tokenId	external

onERC721Received	address operator address from uint256 tokenId bytes data	external
totalSupply	none	external
transfer	address recipient uint amount	external
decimals	none	external
symbol	none	external
transferFrom	address sender address recipient uint amount	external
allowance	address owner address spender	external
approve	address spender uint value	external
burn	uint256 amount	external
_tokenURI	uint _tokenId uint _balanceOf uint _locked_end uint _value	external
encode	bytes data	internal
toString	uint value	internal
_tokenURI	uint _tokenId uint _balanceOf uint _locked_end uint _value	external
resetNoVoted	uint _tokenId	external
setTeam	address _team	external
setArtProxy	address _proxy	external
tokenURI	uint _tokenId	external
ownerOf	uint _tokenId	public
_balance	address _owner	internal
balanceOf	address _owner	external
getApproved	uint _tokenId	external

isApprovedForAll	address _owner address _operator	external
approve	address _approved uint _tokenId	public
setApprovalForAll	address _operator bool _approved	external
_clearApproval	address _owner uint _tokenId	internal
_isApprovedOrOwner	address _spender uint _tokenId	internal
isApprovedOrOwner	address _spender uint _tokenId	external
_transferFrom	address _from address _to uint _tokenId address _sender	internal
transferFrom	address _from address _to uint _tokenId	external
safeTransferFrom	address _from address _to uint _tokenId bytes _data	public
_isContract	address account	internal
supportsInterface	bytes4 _interfaceID	external
tokenOfOwnerByIndex	address _owner uint _tokenIndex	external
_addTokenToOwnerList	address _to uint _tokenId	internal
_addTokenTo	address _to uint _tokenId	internal
_mint	address _to uint _tokenId	internal
_removeTokenFromOwnerList	address _from uint _tokenId	internal
_removeTokenFrom	address _from	internal

	uint _tokenId	
_burn	uint _tokenId	internal
get_last_user_slope	uint _tokenId	external
user_point_history__ts	uint _tokenId uint _idx	external
locked__end	uint _tokenId	external
_checkpoint	uint _tokenId LockedBalance old_locked LockedBalance new_locked	internal
_deposit_for	uint _tokenId uint _value uint unlock_time LockedBalance locked_balance DepositType deposit_type	internal
block_number	none	external
checkpoint	none	external
deposit_for	uint _tokenId uint _value	external
_create_lock	uint _value uint _lock_duration address _to	internal
create_lock	uint _value uint _lock_duration	external
create_lock_for	uint _value uint _lock_duration address _to	external
increase_amount	uint _tokenId uint _value	external
increase_unlock_time	uint _tokenId uint _lock_duration	external
withdraw	uint _tokenId	external
withdrawForce	uint _tokenId	external
withdrawForceCalculate	uint _tokenId	public
split	uint256 _from uint256 _amount	external

_createSplitNFT	address _to LockedBalance _newLocked uint originCreateLockTime	private
_find_block_epoch	uint _block uint max_epoch	internal
_balanceOfNFT	uint _tokenId uint _t	internal
balanceOfNFT	uint _tokenId	external
balanceOfNFTAt	uint _tokenId uint _t	external
_balanceOfAtNFT	uint _tokenId uint _block	internal
balanceOfAtNFT	uint _tokenId uint _block	external
totalSupplyAt	uint _block	external
_supply_at	Point point uint t	internal
totalSupply	none	external
totalSupplyAtT	uint t	public
setVoter	address _voter	external
voting	uint _tokenId	external
abstain	uint _tokenId	external
merge	uint _from uint _to	external

RoguexFactory Contract

Name	Parameter	Attributes
createHypervisor	address tokenA address tokenB uint24 fee string name string symbol	external

setSpotThres	uint256 _spotThres	onlyOwner
setLiqdThres	uint256 _liqdThres	onlyOwner
setPerpThres	uint256 _perpThres	onlyOwner
setPerpRouter	address _router bool _status	onlyOwner
setNftRouter	address _router bool _status	onlyOwner
setSpotRouter	address _router bool _status	onlyOwner
setPoolDeployer	address _dep address _depTrade address _depPos	onlyOwner
setUtils	address _rUtils address _weth	onlyOwner
setHypervisorFactory	address _hypervisorFactory	onlyOwner
createPool	address tokenA address tokenB uint24 fee	external
setOwner	address _owner	external
enableFeeAmount	uint24 fee int24 tickSpacing	public

4. Audit details

4.1 Findings Summary

Severity	Found	Resolved	Acknowledged
● High	0	**0**	0
● Medium	1	0	1
● Low	2	0	2
● Info	4	0	4

4.2 Risk distribution

Name	Risk level	Repair status
Administrator Permissions	Low	Acknowledged
Reentry attack	Info	Acknowledged
Redundant codes	Info	Acknowledged
Logic design flaw	Info	Acknowledged
Price Control	No	normal
Floating Point and Numeric Precision	No	normal
Variables are updated	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
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**

Contract administrator privileges are people who can change the status of a contract or perform certain sensitive operations. If administrator privileges are abused or hacked, the contract may be subject to several risks. First, a hacker may be able to obtain the administrator's private key, thus hijacking the administrator's account and thus having full control of the contract. The hacker can change the status of the contract or perform illegal operations at will, causing a serious loss of contract assets. Second, the administrator may intentionally or unintentionally disclose his or her private key, causing others to gain access. Hackers may obtain the administrator's private key through social engineering or trickery, and obtain a large amount of sensitive contract information, which can then be used to carry out attacks. Third, administrators may use their privileges to perform improper operations. For example, an administrator may change the status of a contract for financial gain, or change the execution rules of a contract to gain additional control, and so on. These actions may also lead to loss of contract assets or illegitimate domination.

1. Some of the parameters in the RoguexFactory contract that can only be modified by the administrator are subject to the risk of centralization, which may pose a risk to the normal operation of the contract if the private key of the owner of the privilege is leaked or if the owner behaves in a malicious manner.

   Risk level：Low

```
function setSpotThres(uint256 _spotThres) external onlyOwner{
    require(_spotThres <= 1000);
    spotThres = _spotThres;
}
function setLiqdThres(uint256 _liqdThres) external onlyOwner{
    require(_liqdThres <= 1000);
    liqdThres = _liqdThres;
}
function setPerpThres(uint256 _perpThres) external onlyOwner{
    require(_perpThres <= 1000);
    perpThres = _perpThres;
}
function setPerpRouter(address _router, bool _status) external
onlyOwner {
    approvedPerpRouters[_router] = _status;
}
function setNftRouter(address _router, bool _status) external
onlyOwner{
    approvedNftRouters[_router] = _status;
}
function setSpotRouter(address _router, bool _status) external
onlyOwner{
    approvedSpotRouters[_router] = _status;
}
function setPoolDeployer(address _dep, address _depTrade, address
_depPos) external onlyOwner{
    spotPoolDeployer = _dep;
    perpPoolDeployer = _depTrade;
    posnPoolDeployer = _depPos;
}
function setUtils(address _rUtils, address _weth) external
onlyOwner{
    utils = _rUtils;
    weth = _weth;
}
function setHypervisorFactory(address _hypervisorFactory) external
onlyOwner{
    hypervisorFactory = _hypervisorFactory;
}
```

2. The `setSwapMining` and `setUtils` methods in the contract allow for modification of key configurations of the contract, but their access control is based on `IRoguexFactory(factory).owner()`, which may pose a centralization risk. Both

functions have `TEST ONLY` annotations, and it is assumed that the function does not exist in subsequent production, so this is just a reminder of the issue, but there is a problem here, if the function does not exist in production, there may be uninitialized issues that can make the contract functionality unavailable.

Risk level：Low

```
function setSwapMining(address addr) public {//TEST ONLY
    require(msg.sender == IRoguexFactory(factory).owner(), "ol");
    swapMining = addr;
}
function setUtils(address _roguUtils, address _perpOrderbook)
external { //TEST ONLY
    require(msg.sender == IRoguexFactory(factory).owner(), "ol");
    roxUtils = _roguUtils;
    perpOrderbook = _perpOrderbook;
}
```

3. The `setTime` and `setFactor` methods in the contract allow for the modification of important parameters in the contract, the access control of which is based on the owner of the factory contract, which may pose a centralization risk.

   Risk level：Low

```
function setTime(uint32 _time) external {
    require(msg.sender == IRoguexFactory(factory).owner(), "f-
owner");
    twapTime = _time;
}
function setFactor(uint256 _kMax, uint256 _powF, uint32 _countMin)
external {
    require(msg.sender == IRoguexFactory(factory).owner(), "f-
owner");
    countMin = _countMin;
    uint256 fs = 100**_powF;
    cFt = CloseFactor({
        kMax: _kMax,
        powF : _powF,
        factor_s : fs,
        factor_sf: (fs)**_powF
    });
    // kMax = _kMax;
    // powF = _powF;
}
```

4. In the contract, the administrator has the right to mint coins and can specify any number of coins to be minted. If this role is owned by the EOA address, there is the possibility that the administrator's authority is too large, which may lead to an increase in the issuance of tokens due to the leakage of the private key or internal manipulation, thus destroying the market.

   Risk level：Medium

```
function mint(address account, uint amount) external returns (bool)
{
    require(msg.sender == minter, "not allowed");
    _mint(account, amount);
    return true;
}
```

- **Safety advice**

Administrator addresses avoid using a single EOA address for control, and try to use more decentralized permissions management, such as multi-signature wallets or governance voting.

- **Repair Status**

ROGUEX has Acknowledged.

4.3.2 Reentry Attack

- **Risk Description**

An attacker constructs a contract containing malicious code at an external address in the Fallback function When the contract sends tokens to this address, it will call the malicious code. The call.value() function in Solidity will consume all the gas he receives when it is used to send tokens, so a re-entry attack will occur when the call to the call.value() function to send tokens occurs before the actual reduction of the sender's account balance. The re-entry vulnerability led to the famous The DAO attack.

In RoxPerpPool contracts, since the transfer function uses the safeTransfer transfer function, which uses a call function, there may be a call to an external contract address that could result in a contract reentry. However, in the current version of the code audit, the contract called the function before the correct update of the position data, so there is no re-entry point of utilization, just a reminder.

```
function _transferOut0(uint256 _amount0, address _recipient, bool
_toETH) private {
    if (_amount0 > 0){
        if (_toETH && token0 == weth){
            IWETH9(weth).withdraw(_amount0);
            TransferHelper.safeTransferETH(_recipient, _amount0);
        }else{
            TransferHelper.safeTransfer(token0, _recipient,
_amount0);
        }
        sBalance0 = balance0();
    }
}

function _transferOut1(uint256 _amount1, address _recipient, bool
_toETH) private {
    if (_amount1 > 0){
        if (_toETH && token1 == weth){
            IWETH9(weth).withdraw(_amount1);
            TransferHelper.safeTransferETH(_recipient, _amount1);
        }else{
            TransferHelper.safeTransfer(token1, _recipient,
_amount1);
        }
        sBalance1 = balance1();
    }
}
```

- **Safety advice**

Three ways to mitigate this problem: 1. Ensure that all transfer operations are executed after a state change so that they do not satisfy the reentry condition; 2. Add reentry locks to external functions in the contract that use both functions; and 3. Limit the relevant contract addresses, prohibit contract calls, and add a whitelist of non-contract addresses.

- **Repair Status**

ROGUEX has Acknowledged.

4.3.3 Redundant Codes

- **Risk description**

The RoxUtils contract has the error message mapping `errMsg` declared but doesn't seem to add a specific mapping as well as use it, which could be a waste of extra deployment gas costs.

- **Safety advice**

Adjusted with the needs of the project side, the code is deleted if it is not required to be used, and if it is required to be used, the content of the initialized variable needs to be confirmed.

- **Repair Status**

ROGUEX has Acknowledged.

4.3.4 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.

In both ROXToken and VotingEscrow contracts, there is the problem that some of the logic may be commented out due to testing, and the code does not check the caller, so there may be the risk of unchecked permissions.

```
//TODO:
function setVoter(address _voter) external {
    // require(msg.sender == voter);
    voter = _voter;
}
//TODO:
function setMinter(address _minter) external {
    // require(msg.sender == minter);
    minter = _minter;
}
```

- **Safety advice**

It needs to need to be restored at the time of formal production deployment, otherwise there could be serious security issues.

- **Repair Status**

ROGUEX has Acknowledged.

4.3.5 Price Control

- **Risk description**

Price manipulation usually refers to the practice of some large investors or traders of buying or selling large quantities of a particular asset to influence the price of that asset and then capitalizing on the price change to make a profit. This behavior undermines the fairness of the market and creates an uneven playing field for smaller investors.

- **Audit Results : Passed**

4.3.6 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**

4.3.7 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**

4.3.8 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 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**

4.3.17 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**

4.3.18 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-passte address instead of the standard 20-passte 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**

4.3.19 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**

4.3.20 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**

4.3.21 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**

4.3.22 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**

1. 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.

LUNARAY
BLOCKCHAINSECURITY

https://lunaray.co

https://github.com/lunaraySec

https://twitter.com/lunaray_co

http://t.me/lunaraySec