Cyberscope

Audit Report
LuncWarrior

September 2022

Type	BEP20
Network	BSC
Address	0xad4322C5616F22bcCF51d6B735c57F902922d955
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	LuncWarrior
Compiler Version	v0.8.15+commit.e14f2714
Optimization	200 runs
Licence	MIT
Explorer	https://bscscan.com/token/0xad4322C5616F22bcCF51d6B735c57F902922d955
Symbol	LUNAR
Decimals	9
Total Supply	100,000,000,000
Domain	https://luncwarrior.com

Source Files

Filename	**SHA256**
contract.sol	33c2d96a0e4a1db492a37eeb00ac03777ce377aff29a914fbc2a104652d20bbf

Audit Updates

Initial Audit	12th September 2022
Corrected	

Contract Analysis

🔴 Critical 🟠 Medium ⚪ Minor / Informative 🔵 Pass

Severity	Code	Description	Status
🔵	ST	Stops Transactions	Unresolved
⚪	OCTD	Transfers Contract's Tokens	Unresolved
🔵	OTUT	Transfers User's Tokens	Passed
🔵	ELFM	Exceeds Fees Limit	Passed
⚪	ULTW	Transfers Liquidity to Team Wallet	Unresolved
🔵	MT	Mints Tokens	Passed
🔵	BT	Burns Tokens	Passed
🔵	BC	Blacklists Addresses	Passed

OCTD - Transfers Contract's Tokens

Criticality	minor / informative
Location	contract.sol#L1108
Status	Unresolved

Description

The contract owner has the authority to claim all the balance of the contract. The owner may take advantage of it by calling the `transferForeignToken` function.

```
function transferForeignToken(address _token, address _to, uint256 _value) external
onlyOwner returns(bool _sent){
        if(_value == 0) {
            _value = IERC20(_token).balanceOf(address(this));
        }
        _sent = IERC20(_token).transfer(_to, _value);
    }
```

Recommendation

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

ULTW - Transfers Liquidity to Team Wallet

Criticality	minor / informative
Location	contract.sol#L1115
Status	Unresolved

Description

The contract owner has the authority to transfer funds without limit to the team wallet. These funds have been accumulated from fees collected from the contract. The owner may take advantage of it by calling the Sweep methods.

```
function Sweep() external onlyOwner {
        uint256 balance = address(this).balance;
        payable(owner()).transfer(balance);
    }
```

Recommendation

The contract could embody a check for the maximum amount of funds that can be swapped. Since a huge amount may volatile the token’s price.

The team should carefully manage the private keys of the owner’s account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

Contract Diagnostics

🔴 Critical 🟠 Medium ⚪ Minor / Informative

Severity	Code	Description	Status
🟠	BLC	Business Logic Concern	Unresolved
🟠	US	Untrusted Source	Unresolved
⚪	STC	Succeeded Transfer Check	Unresolved
⚪	CO	Code Optimization	Unresolved
⚪	L01	Public Function could be Declared External	Unresolved
⚪	L02	State Variables could be Declared Constant	Unresolved
⚪	L03	Redundant Statements	Unresolved
⚪	L04	Conformance to Solidity Naming Conventions	Unresolved
⚪	L05	Unused State Variable	Unresolved
⚪	L07	Missing Events Arithmetic	Unresolved
⚪	L09	Dead Code Elimination	Unresolved
⚪	L11	Unnecessary Boolean equality	Unresolved
⚪	L12	Using Variables before Declaration	Unresolved

●	L13	Divide before Multiply Operation	Unresolved
●	L14	Uninitialized Variables in Local Scope	Unresolved
●	L15	Local Scope Variable Shadowing	Unresolved

BLC - Business Logic Concern

Criticality	medium
Location	contract.sol#L1181
Status	Unresolved

Description

The business logic seems peculiar. The implementation may not follow the expected behavior.

The following fee percentages of marketingBnb, lunaBurnBnb, liqBnb are not obtained from the initiale value of contracts’s balance.

```
uint256 balancez = address(this).balance;
if(marketingSwapSendActive && marketingSellFee > 0) {
    uint256 marketingBnb = balancez.mul(marketingSellFee).div(totalSellFees);
    (bool success,) = address(marketingWallet).call{value: marketingBnb}("");
    if(success) {emit MarketingFeeCollected(marketingBnb);}
    balancez -= marketingBnb;
}
if(lunaBurnSwapSendActive  && LUNABurnSellFee > 0) {
    uint256 lunaBurnBnb = balancez.mul(LUNABurnSellFee).div(totalSellFees);
    (bool success,) = address(lunaBurnWallet).call{value: lunaBurnBnb}("");
    if(success) {emit LunaBurnFeeCollected(lunaBurnBnb);}
    balancez -= lunaBurnBnb;
}
if(LiqSwapSendActive){
    uint256 liqBnb = balancez.mul(liqSellFee).div(totalSellFees);
    if(liqBnb > 5) { // failsafe if addLiq is too low
        addLiquidity(liqPart, liqBnb);
        balancez -= liqBnb;
```

Recommendation

The team is advised to carefully check if the implementation follows the expected business logic.

US - Untrusted Source

Criticality	medium
Location	contract.sol#L857
Status	Unresolved

Description

The contract uses an external contract in order to determine the transaction's flow. The external contract is untrusted. As a result it may produce security issues and harm the transactions.

```
_LUNADividendTracker public _lunaDividendTracker;
```

Recommendation

The contract should use a trusted external source. A trusted source could be either a commonly recognized or an audited contract. The pointing addresses should not be able to change after the initialization.

STC - Succeeded Transfer Check

Criticality	minor / informative
Location	contract.sol#L1112
Status	Unresolved

Description

According to the ERC20 specification, the transfer methods should be checked if the result is successful. Otherwise, the contract may wrongly assume that the transfer has been established.

```
_sent = IERC20(_token).transfer(_to, _value);
```

Recommendation

The contract should check if the result of the transfer methods is successful.

CO - Code Optimization

Criticality	minor / informative
Location	contract.sol#L909
Status	Unresolved

Description

There are code segments that could be optimized. A segment may be optimized so that it becomes a smaller size, consumes less memory, executes more rapidly, or performs fewer operations.

This code segment can be optimized. The structure userData is redundant since it has only one variable.

```
struct userData {
        uint lastBuyTime;
}
```

Recommendation

Rewrite some code segments so the runtime will be more performant.

L01 - Public Function could be Declared External

Criticality	minor / informative
Location	contract.sol#L744,133,50,1115,294,144,103,119,1321,139,55,788,254,99,267,1132,1121,128,302,1261,124
Status	Unresolved

Description

Public functions that are never called by the contract should be declared external to save gas.

```
getAccountAtIndex
transferFrom
renounceOwnership
edit_excludeFromFees
dividendOf
decreaseAllowance
symbol
transfer
processDividendTracker
...
```

Recommendation

Use the external attribute for functions never called from the contract.

L02 - State Variables could be Declared Constant

Criticality	minor / informative
Location	contract.sol#L857,228,860
Status	Unresolved

Description

Constant state variables should be declared constant to save gas.

```
lunaBurnWallet
lastAmount
deadWallet
```

Recommendation

Add the constant attribute to state variables that never change.

L03 - Redundant Statements

Criticality	minor / informative
Location	contract.sol#L19
Status	Unresolved

Description

The contract contains statements that are not used and have no effect. As a result, those segments increase the code size of the contract unnecessarily.

```
Context
```

Recommendation

Remove the redundant statements in order to decrease the code size.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor / informative
Location	contract.sol#L1104,1141,1096,905,1291,890,1238,1044,1227,1152,1151,859,663,944,1052,1076,1150,1071,865,1216,1221,379,1235,396,1169,1326,225,888,1111,1297,626,378,699,1288,889,922,1038,294,1100,887,302,1335,853,298,235,916,424,1057,1010,307,1341,1294,867,1309,1115,868
Status	Unresolved

Description

Solidity defines a naming convention that should be followed. Rule exceptions:

- Allow constant variable name/symbol/decimals to be lowercase.
- Allow _ at the beginning of the mixed_case match for private variables and unused parameters.

```
_token
_states
_value
userData
withdrawable_LUNADividendOf
LUNABurnSellFee
update_lunaDividendTracker
_address
_intervalSecondsForSwap
...
```

Recommendation

Follow the Solidity naming convention.
https://docs.soliditylang.org/en/v0.4.25/style-guide.html#naming-conventions.

L05 - Unused State Variable

Criticality	minor / informative
Location	contract.sol#L228
Status	Unresolved

Description

There are segments that contain unused state variables.

```
lastAmount
```

Recommendation

Remove unused state variables.

L07 - Missing Events Arithmetic

Criticality	minor / informative
Location	contract.sol#L1150,1082,1132,1044,1076
Status	Unresolved

Description

Detected missing events for critical arithmetic parameters. There are functions that have no event emitted, so it is difficult to track off-chain changes.

```
LUNARewardsBuyFee = _reward_buy
buySecondsLimit = buy
maxWalletAmount = max * 10 ** decimals()
intervalSecondsForSwap = _intervalSecondsForSwap
maxBuyTxAmount = _buy * 10 ** decimals()
```

Recommendation

Emit an event for critical parameter changes.

L09 - Dead Code Elimination

Criticality	minor / informative
Location	contract.sol#L453,188,312
Status	Unresolved

Description

Functions that are not used in the contract, and make the code's size bigger.

```
get
_setupDecimals
_transfer
```

Recommendation

Remove unused functions.

L11 - Unnecessary Boolean equality

Criticality	minor / informative
Location	contract.sol#L1085
Status	Unresolved

Description

The comparison to boolean constants is redundant. Boolean constants can be used directly and do not need to be compared to true or false.

```
require(bool)(marketActive == false)
```

Recommendation

Remove the equality to the boolean constant.

L12 - Using Variables before Declaration

Criticality	minor / informative
Location	contract.sol#L1431,1190
Status	Unresolved

Description

The contract is using a variable before the declaration. This is usually happening either if it has not been declared yet or the variable has been declared in a different scope.

```
iterations
success
claims
lastProcessedIndex
```

Recommendation

The variables should be declared before any usage of them.

L13 - Divide before Multiply Operation

Criticality	minor / informative
Location	contract.sol#L1346
Status	Unresolved

Description

Performing divisions before multiplications may cause lose of prediction.

```
txFees_scope_0 = amount * totalSellFees / 100
txFees = amount * totalBuyFees / 100
```

Recommendation

The multiplications should be prior to the divisions.

L14 - Uninitialized Variables in Local Scope

Criticality	minor / informative
Location	contract.sol#L1431,1196
Status	Unresolved

Description

The are variables that are defined in the local scope and are not initialized.

```
lastProcessedIndex
claims
success_scope_0
iterations
```

Recommendation

All the local scoped variables should be initialized.

L15 - Local Scope Variable Shadowing

Criticality	minor / informative
Location	contract.sol#L244,307,298,1052,294,302
Status	Unresolved

Description

The are variables that are defined in the local scope containing the same name from an upper scope.

```
_name
_owner
_symbol
_burn
```

Recommendation

The local variables should have different names from the upper scoped variables.

Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
ERC20	Implementation	Context, IERC20		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-

	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_setupDecimals	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
IDividendPayingToken	Interface			
	dividendOf	External		-
	withdrawDividend	External	✓	-
IDividendPayingTokenOptional	Interface			
	withdrawableDividendOf	External		-
	withdrawnDividendOf	External		-
	accumulativeDividendOf	External		-
DividendPayingToken	Implementation	ERC20, IDividendPayingToken, IDividendPayingTokenOptional, Ownable		
	<Constructor>	Public	✓	ERC20
	setAuth	External	✓	onlyOwner
	distributeDividends	Public	✓	onlyOwner
	withdrawDividend	Public	✓	-
	setDividendTokenAddress	External	✓	onlyOwner
	_withdrawDividendOfUser	Internal	✓	
	dividendOf	Public		-
	withdrawableDividendOf	Public		-
	withdrawnDividendOf	Public		-

	accumulativeDividendOf	Public		-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_setBalance	Internal	✓	
IUniswapV2Factory	Interface			
	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-
IUniswapV2Pair	Interface			
	name	External		-
	symbol	External		-
	decimals	External		-
	totalSupply	External		-
	balanceOf	External		-
	allowance	External		-
	approve	External	✓	-
	transfer	External	✓	-
	transferFrom	External	✓	-
	DOMAIN_SEPARATOR	External		-
	PERMIT_TYPEHASH	External		-
	nonces	External		-
	permit	External	✓	-
	MINIMUM_LIQUIDITY	External		-
	factory	External		-
	token0	External		-
	token1	External		-

	getReserves	External		-
	price0CumulativeLast	External		-
	price1CumulativeLast	External		-
	kLast	External		-
	mint	External	✓	-
	burn	External	✓	-
	swap	External	✓	-
	skim	External	✓	-
	sync	External	✓	-
	initialize	External	✓	-
IUniswapV2Router01	Interface			
	addLiquidityETH	External	Payable	-
	factory	External		-
	WETH	External		-
	swapExactTokensForTokens	External	✓	-
	swapTokensForExactTokens	External	✓	-
	swapExactETHForTokens	External	Payable	-
	swapTokensForExactETH	External	✓	-
	swapExactTokensForETH	External	✓	-
	swapETHForExactTokens	External	Payable	-
IUniswapV2Router02	Interface	IUniswapV2Router01		
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
IterableMapping	Library			

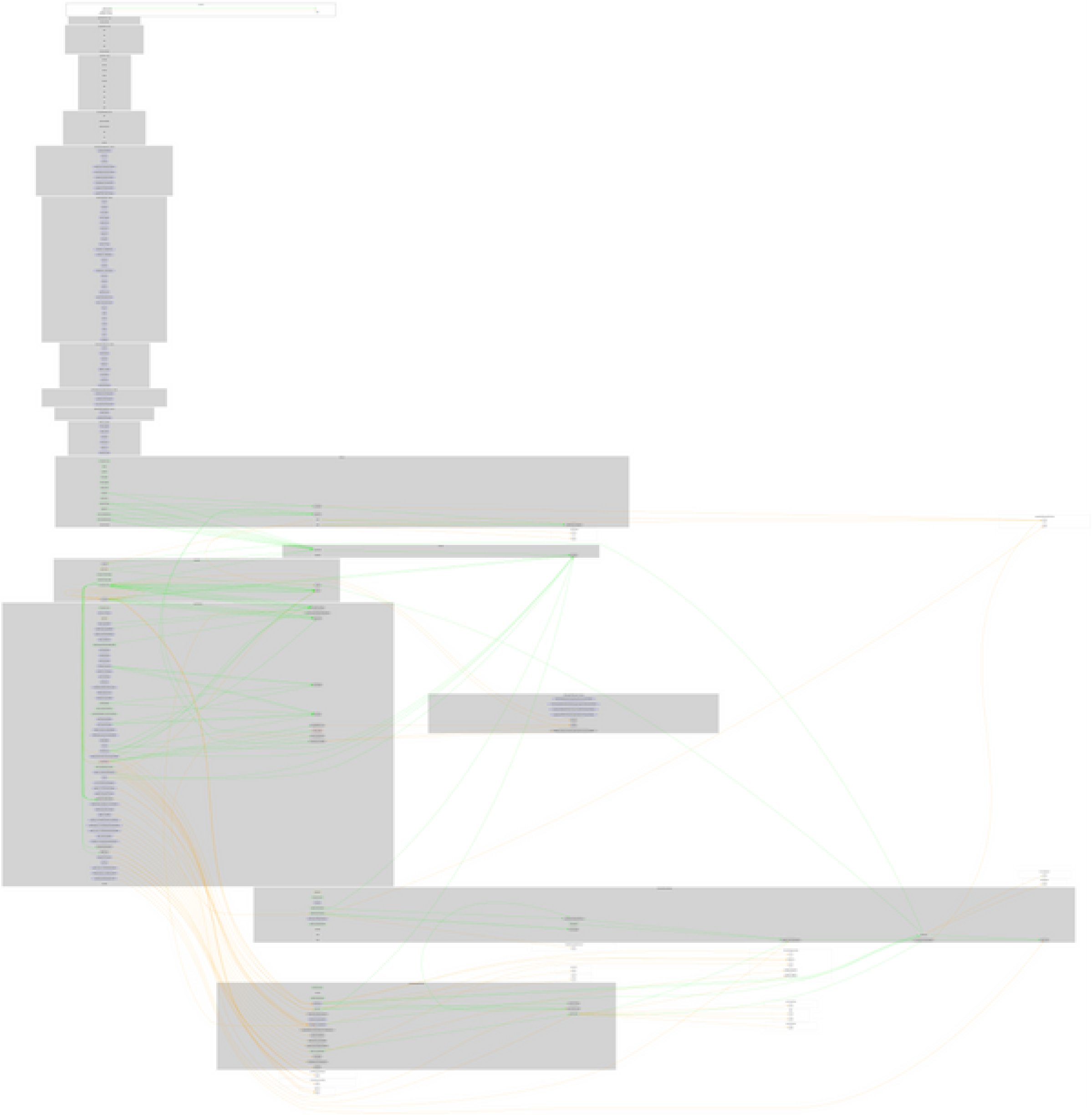
	get	Internal		
	getIndexOfKey	Internal		
	getKeyAtIndex	Internal		
	size	Internal		
	set	Internal	✓	
	remove	Internal	✓	
SafeMath	Library			
	tryAdd	Internal		
	trySub	Internal		
	tryMul	Internal		
	tryDiv	Internal		
	tryMod	Internal		
	add	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	mod	Internal		
	sub	Internal		
	div	Internal		
	mod	Internal		
SafeMathInt	Library			
	mul	Internal		
	div	Internal		
	sub	Internal		
	add	Internal		
	toUint256Safe	Internal		
SafeMathUint	Library			
	toInt256Safe	Internal		
_LUNADividendTracker	Implementation	DividendPayingToken		
	<Constructor>	Public	✓	DividendPayingToken

	_transfer	Internal		
	withdrawDividend	Public		-
	setDividendTokenAddress	External	✓	onlyOwner
	updateMinimumTokenBalanceForDividends	External	✓	onlyOwner
	excludeFromDividends	External	✓	onlyOwner
	includeFromDividends	External	✓	onlyOwner
	updateClaimWait	External	✓	onlyOwner
	getLastProcessedIndex	External		-
	getNumberOfTokenHolders	External		-
	getAccount	Public		-
	getAccountAtIndex	Public		-
	canAutoClaim	Private		
	setBalance	External	✓	onlyOwner
	process	Public	✓	-
	processAccount	Public	✓	onlyOwner
LuncWarrior	Implementation	ERC20, Ownable		
	<Constructor>	Public	✓	ERC20
	<Receive Ether>	External	Payable	-
	KKPunish	Private	✓	
	prepareForLaunch	External	✓	onlyOwner
	setProcessDividendStatus	External	✓	onlyOwner
	setLunaAddress	External	✓	onlyOwner
	setSwapAndLiquify	External	✓	onlyOwner
	setSwapSend	External	✓	onlyOwner
	setMultiBlock	External	✓	onlyOwner
	addLiquidity	Private	✓	
	setFeesDetails	External	✓	onlyOwner
	setMaxTxAmount	External	✓	onlyOwner
	setBuySecondLimits	External	✓	onlyOwner
	activateMarket	External	✓	onlyOwner
	editLimits	External	✓	onlyOwner
	setMinimumWeiForTokenomics	External	✓	onlyOwner

	editPreMarketUser	External	✓	onlyOwner
	transferForeignToken	External	✓	onlyOwner
	Sweep	External	✓	onlyOwner
	edit_excludeFromFees	Public	✓	onlyOwner
	excludeMultipleAccountsFromFees	Public	✓	onlyOwner
	setMarketingWallet	External	✓	onlyOwner
	setMaxWallet	Public	✓	onlyOwner
	editExcludedFromMaxWallet	External	✓	onlyOwner
	editMultiExcludedFromMaxWallet	External	✓	onlyOwner sameSize
	setliqWallet	External	✓	onlyOwner
	setFees	External	✓	onlyOwner
	KKAirdrop	External	✓	onlyOwner
	swapTokens	Private	✓	
	prepareForPartherOrExchangeListing	External	✓	onlyOwner
	updateMarketingWallet	External	✓	onlyOwner
	updateLiqWallet	External	✓	onlyOwner
	setAuthOnDividends	Public	✓	onlyOwner
	set_LUNADividendEnabled	External	✓	onlyOwner
	update_lunaDividendTracker	External	✓	onlyOwner
	updateUniswapV2Router	External	✓	onlyOwner
	excludeFromFees	Public	✓	onlyOwner
	excludeFromDividend	Public	✓	onlyOwner
	setAutomatedMarketMakerPair	Public	✓	onlyOwner
	_setAutomatedMarketMakerPair	Private	✓	onlyOwner
	updateGasForProcessing	External	✓	onlyOwner
	updateMinimumBalanceForDividends	External	✓	onlyOwner
	updateClaimWait	External	✓	onlyOwner
	getLUNAClaimWait	External		-
	getTotal_LUNADividendsDistributed	External		-
	withdrawable_LUNADividendOf	External		-
	_lunaDividendTokenBalanceOf	External		-
	getAccount_LUNADividendsInfo	External		-
	getAccount_LUNADividendsInfoAtIndex	External		-

	processDividendTracker	Public	✓	onlyOwner
	update_LUNADividendToken	External	✓	onlyOwner
	claim	External	✓	-
	getLast_LUNADividendProcessedIndex	External		-
	getNumberOf_LUNADividendTokenHolders	External		-
	_transfer	Internal	✓	
	swapTokensForBNB	Private	✓	
	swapBNBforLuna	Private	✓	
	transferDividends	Private	✓	

Contract Flow



Domain Info

Domain Name	https://luncwarrior.com
Registry Domain ID	2722630624_DOMAIN_COM-VRSN
Creation Date	2022-09-02T09:13:04.00Z
Updated Date	0001-01-01T00:00:00.00Z
Registry Expiry Date	2023-09-02T09:13:04.00Z
Registrar WHOIS Server	whois.namecheap.com
Registrar URL	http://www.namecheap.com
Registrar	NAMECHEAP INC
Registrar IANA ID	1068

The domain was created 10 days before the creation of the audit. It will expire in 12 months.

There is no public billing information, the creator is protected by the privacy settings.

Summary

There are some functions that can be abused by the owner like transferring tokens to the team's wallet and transferring funds to the team's wallet. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats. There is also a limit of max 12,5% fees.

Disclaimer

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment.

Cyberscope team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed.

The Cyberscope team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Cyberscope receive a payment to manipulate those results or change the awarding badge that we will be adding in our website.

Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token.

The Cyberscope team disclaims any liability for the resulting losses.

About Cyberscope

Coinscope audit and K.Y.C. service has been rebranded to Cyberscope.

Coinscope is the leading early coin listing, voting and auditing authority firm. The audit process is analyzing and monitoring many aspects of the project. That way, it gives the community a good sense of security using an informative report and a generic score.

Cyberscope and Coinscope are aiming to make crypto discoverable and efficient globally. They provide all the essential tools to assist users draw their own conclusions.

The Cyberscope team

https://www.cyberscope.io