Cyberscope

Audit Report

ETH SHIBA

May 2022

Type	BEP20
Network	BSC
Address	0x69d10c8Bd0de1a9345AFA36819490D8BbCE0E5A3
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	ETHSHIB
Compiler Version	v0.8.0+commit.c7dfd78e
Optimization	200 runs
Licence	None
Explorer	https://bscscan.com/token/0x69d10c8Bd0de1a9345AFA36819490D8BbCE0E5A3
Symbol	$ETHSHIB
Decimals	18
Total Supply	1,000,000,000,000,000
Domain	ethshiba.io

Source Files

Filename	**SHA256**
contract.sol	674ddb10cd99d9422a760105d60a381e1dc2e36bd63b9064dc028089bed2da13

Audit Updates

Initial Audit	23rd May 2022
Corrected	

Contract Analysis

Critical Medium Minor Pass

Severity	Code	Description
●	ST	Contract Owner is not able to stop or pause transactions
●	OCTD	Contract Owner is not able to transfer tokens from specific address
●	OTUT	Owner Transfer User's Tokens
●	ELFM	Contract Owner is not able to increase fees more than a reasonable percent (25%)
●	ULTW	Contract Owner is not able to increase the amount of liquidity taken by dev wallet more than a reasonable percent
●	MT	Contract Owner is not able to mint new tokens
●	BT	Contract Owner is not able to burn tokens from specific wallet
●	BC	Contract Owner is not able to blacklist wallets from selling

ST - Stop Transactions

Criticality	critical
Location	contract.sol#L 1598, 1614, 1661, 1572

Description

The contract owner has the authority to stop transactions for all users excluding the owner. The owner may take advantage of it by setting the maxWalletToken and/or maxTransactionAmount to zero.

```
require(
    contractBalanceRecepient + amount <= maxWalletToken,
    "Exceeds maximum wallet token amount."
);
```

```
require(amount <= maxTransactionAmount, "transfer amount exceeds the
maxSellTransactionAmount.");
```

The contract owner can also convert the contract into a honeypot and prevent the users from selling by setting the _SellTotalFees a higher value than _BuyTotalFees or if the return value of the extraFeeOnSell function is over 100.

```
if(takeFee) {
           uint256 fees = amount.mul(_BuyTotalFees).div(100);
            if(automatedMarketMakerPairs[to]){
                fees += amount.mul(extraFeeOnSell()).div(100);
            }
            amount = amount.sub(fees);
```

```
function extraFeeOnSell() internal view returns(uint256)
    {
        return _SellTotalFees.sub(_BuyTotalFees);
    }
```

Recommendation

The contract could embody a check for not allowing setting the maxWalletAmount and maxTransactionAmount less than a reasonable amount. A suggested implementation could check that the minimum amount should be more than a fixed percentage of the total supply.

The contract could embody a check for not allowing setting the taxes more than a reasonable amount.

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.

ELFM - Exceed Limit Fees Manipulation

Criticality	critical
Location	contract.sol#L1451, 1461

Description

The contract owner has the authority to increase over the allowed limit of 25%. The owner may take advantage of it by calling the `setBuyFees` and `setSellFees` functions with a high percentage value.

```
function setBuyFees(uint256 buyEthRewardFee, uint256 buyLiquidityFee, uint256
buyMarketingFee, uint256 buyBackSHIBAFee) external onlyOwner
    {
        _BuyETHRewardsFee = buyEthRewardFee;
        _BuyLiquidityFee = buyLiquidityFee;
        _BuyMarketingFee = buyMarketingFee;
        _BuyBackSHIBAFee = buyBackSHIBAFee;
        _BuyTotalFees =
_BuyETHRewardsFee.add(_BuyLiquidityFee).add(_BuyMarketingFee).add(_BuyBackSHIB
AFee);

    }
```

```
function setSellFees(uint256 sellEthRewardFee, uint256 sellLiquidityFee,
uint256 sellMarketingFee, uint256 sellBackSHIBAFee) external onlyOwner
    {
        _SellETHRewardsFee = sellEthRewardFee;
        _SellLiquidityFee = sellLiquidityFee;
        _SellMarketingFee = sellMarketingFee;
        _SellBackSHIBAFee = sellBackSHIBAFee;
        _SellTotalFees =
_SellETHRewardsFee.add(_SellLiquidityFee).add(_SellMarketingFee).add(_SellBack
SHIBAFee);

    }
```

Recommendation

The contract could embody a check for the maximum acceptable value.

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 - Unlimited Liquidity to Team Wallet

Criticality	minor
Location	contract.sol#L1567, 1734, 1804

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 setting a high fee to the swapTokensAtAmount variable and by calling the manualSwapTokensForEth and withdrawStuckBNB functions.

```
function changeSwapTokensAtAmount(uint256 swapAmount) external onlyOwner
    {
        swapTokensAtAmount = swapAmount;
    }
```

```
function manualSwapTokensForEth(uint256 tokenAmount) public onlyOwner {


        // generate the uniswap pair path of token -> weth
        address[] memory path = new address[](2);
        path[0] = address(this);
        path[1] = uniswapV2Router.WETH();

        _approve(address(this), address(uniswapV2Router), tokenAmount);

        // make the swap
        uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
            tokenAmount,
            0, // accept any amount of ETH
            path,
            address(this),
            block.timestamp
        );
```

```
function withdrawStuckBNB() external onlyOwner{
        require (address(this).balance > 0, "Can't withdraw negative or
zero");
        payable(owner()).transfer(address(this).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.

BC - Blacklisted Contracts

Criticality	medium
Location	contract.sol#L1584, 1586

Description

The contract owner has the authority to stop contracts from transactions. The owner may take advantage of it by calling the addToBlacklist function.

```
require(!_blacklisted[to] && !_blacklisted[from], "address is blacklisted");
```

Also, the owner can block users by enabling the tradingEnabed boolean and calling the addToWhitelist function.

```
 if(!_whitelisted[from]) { require(tradingEnabled, "Trading is not enabled
yet");}
```

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.

Contract Diagnostics

Critical Medium Minor

Severity	Code	Description
	L01	Public Function could be Declared External
	L04	Conformance to Solidity Naming Conventions
	L05	Unused State Variable
	L07	Missing Events Arithmetic
	L09	Dead Code Elimination
	L12	Using Variables before Declaration
	L14	Uninitialized Variables in Local Scope
	L15	Local Scope Variable Shadowing

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L352,378,386,397,415,437,456,741,750,800,815,843,857,930,934,941,947,1351,1360,1367,1388,1407,1436,1481,1500,1504,1508,1512,1516,1734,1922,1966

Description

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

```
process
getAccountAtIndex
manualSwapTokensForEth
dividendTokenBalanceOf
withdrawableDividendOf
isExcludedFromMaxWallet
isExcludedFromMaxTx
isExcludedFromFees
updateGasForProcessing
...
```

Recommendation

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

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L843,850,857,867,769,775,998,999,1016,1052,1252,1336,1341,1346,1351,1355,1431,1436,1809,1198,1200,1205,1206,1207,1208,1210,1212,1213,1214,1215,1216,1220,1221,1877

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.

```
_account
_shibaWalletAddress
_marketingWalletAddress
_SellTotalFees
_SellBackSHIBAFee
_SellMarketingFee
_SellLiquidityFee
_SellETHRewardsFee
_BuyTotalFees
...
```

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
Location	contract.sol#L594

Description

There are segments that contain unused state variables.

```
MAX_INT256
```

Recommendation

Remove unused state variables.

L07 - Missing Events Arithmetic

Criticality	minor
Location	contract.sol#L1431,1436,1451,1461,1567

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.

```
swapTokensAtAmount = swapAmount
_SellETHRewardsFee = sellEthRewardFee
_BuyETHRewardsFee = buyEthRewardFee
maxTransactionAmount = _maxTxAmount
maxWalletToken = _maxToken
```

Recommendation

Emit an event for critical parameter changes.

L09 - Dead Code Elimination

Criticality	minor
Location	contract.sol#L877,640

Description

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

```
abs
_transfer
```

Recommendation

Remove unused functions.

L12 - Using Variables before Declaration

Criticality	minor
Location	contract.sol#L1679

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
claims
lastProcessedIndex
```

Recommendation

The variables should be declared before any usage of them.

L14 - Uninitialized Variables in Local Scope

Criticality	minor
Location	contract.sol#L1679

Description

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

```
iterations
lastProcessedIndex
claims
```

Recommendation

All the local scoped variables should be initialized.

L15 - Local Scope Variable Shadowing

Criticality	minor
Location	contract.sol#L795,843,850,857,867

Description

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

```
_owner
_symbol
_name
```

Recommendation

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

Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
IERC20Metadata	Interface	IERC20		
	name	External		-
	symbol	External		-
	decimals	External		-
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
SafeMath	Library			
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
	mod	Internal		
	mod	Internal		

ERC20	Implementation	Context, IERC20, IERC20Metadata		
	<Constructor>	Public	✓	-
	name	External		-
	symbol	External		-
	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	✓	
	_beforeTokenTransfer	Internal	✓	
SafeMathUint	Library			
	toInt256Safe	Internal		
SafeMathInt	Library			
	mul	Internal		
	div	Internal		
	sub	Internal		
	add	Internal		
	abs	Internal		
	toUint256Safe	Internal		
DividendPayingTokenInterface	Interface			
	dividendOf	External		-

	withdrawDividend	External	✓	-
DividendPayingTokenOptionalInterface	Interface			
	withdrawableDividendOf	External		-
	withdrawnDividendOf	External		-
	accumulativeDividendOf	External		-
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
DividendPayingToken	Implementation	ERC20, Ownable, DividendPayingTokenInterface, DividendPayingTokenOptionalInterface		
	<Constructor>	Public	✓	ERC20
	distributeETHDividends	Public	✓	onlyOwner
	withdrawDividend	Public	✓	-
	_withdrawDividendOfUser	Internal	✓	
	dividendOf	Public		-
	withdrawableDividendOf	Public		-
	withdrawnDividendOf	Public		-
	accumulativeDividendOf	Public		-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_setBalance	Internal	✓	
IterableMapping	Library			

	get	Public		-
	getIndexOfKey	Public		-
	getKeyAtIndex	Public		-
	size	Public		-
	set	Public	✓	-
	remove	Public	✓	-
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	✓	-

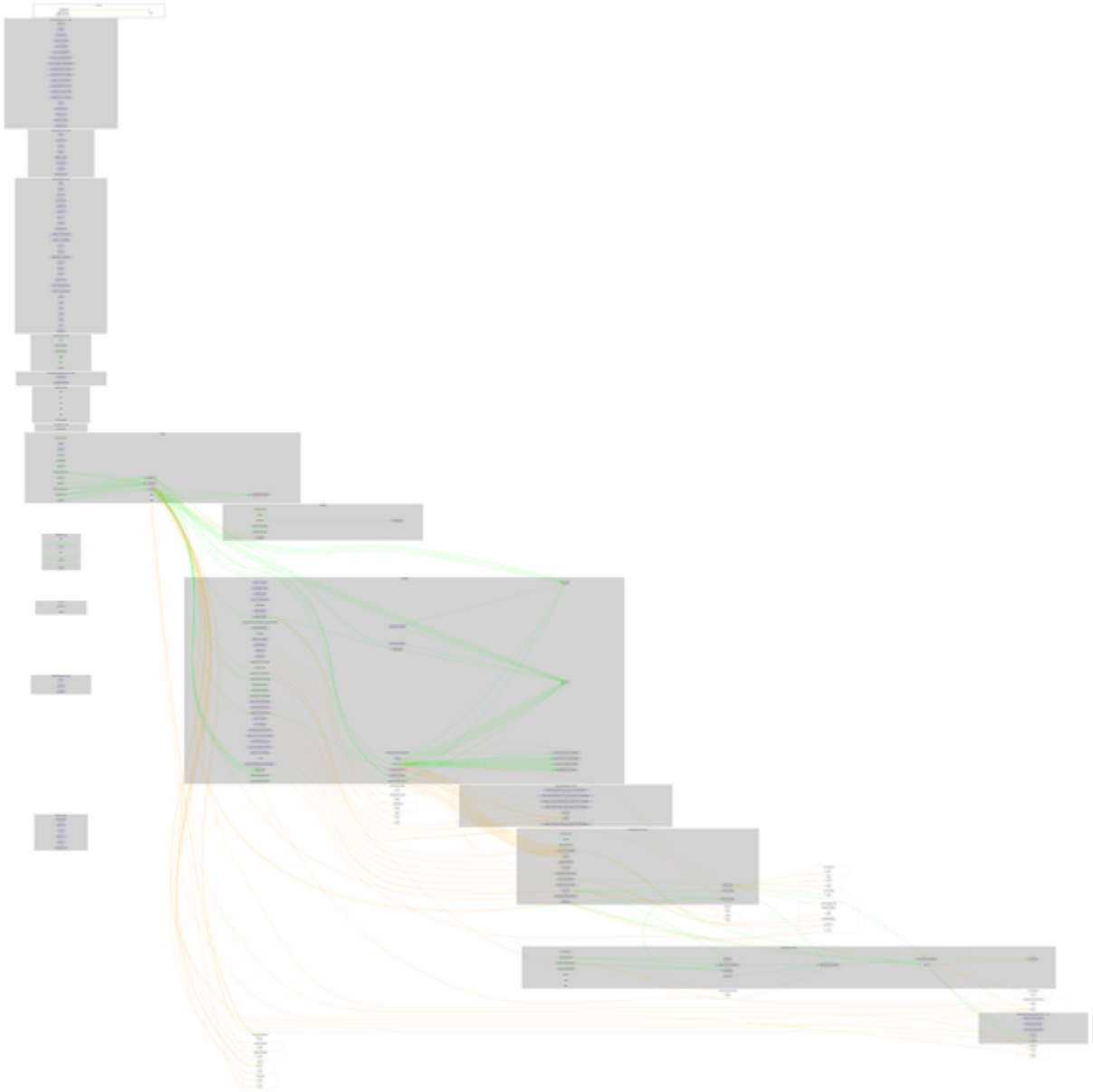
	initialize	External	✓	-
IUniswapV2Factory	Interface			
	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-
IUniswapV2Router01	Interface			
	factory	External		-
	WETH	External		-
	addLiquidity	External	✓	-
	addLiquidityETH	External	Payable	-
	removeLiquidity	External	✓	-
	removeLiquidityETH	External	✓	-
	removeLiquidityWithPermit	External	✓	-
	removeLiquidityETHWithPermit	External	✓	-
	swapExactTokensForTokens	External	✓	-
	swapTokensForExactTokens	External	✓	-
	swapExactETHForTokens	External	Payable	-
	swapTokensForExactETH	External	✓	-
	swapExactTokensForETH	External	✓	-
	swapETHForExactTokens	External	Payable	-
	quote	External		-
	getAmountOut	External		-
	getAmountIn	External		-
	getAmountsOut	External		-
	getAmountsIn	External		-

IUniswapV2Router02	Interface	IUniswapV2Router01		
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
ETHSHIB	Implementation	ERC20, Ownable		
	<Constructor>	Public	✓	ERC20
	<Receive Ether>	External	Payable	-
	setTradingEnabled	External	✓	onlyOwner
	addToBlackList	External	✓	onlyOwner
	removeFromBlackList	External	✓	onlyOwner
	isBlacklisted	Public		-
	addToWhitelist	External	✓	onlyOwner
	setAutomatedMarketMakerPair	Public	✓	onlyOwner
	updateDividendTracker	Public	✓	onlyOwner
	updateUniswapV2Router	Public	✓	onlyOwner
	excludeOrIncludeFromFees	Public	✓	onlyOwner
	excludeOrIncludeMultipleAccountsFromFees	Public	✓	onlyOwner
	excludeOrIncludeFromMaxTx	Public	✓	onlyOwner
	excludeOrIncludeFromMaxWallet	Public	✓	onlyOwner
	setMaxWalletToken	External	✓	onlyOwner
	setMaxtx	Public	✓	onlyOwner
	setMarketingWallet	External	✓	onlyOwner
	setShibaWallet	External	✓	onlyOwner
	setBuyFees	External	✓	onlyOwner
	setSellFees	External	✓	onlyOwner
	_setAutomatedMarketMakerPair	Private	✓	

	updateGasForProcessing	Public	✓	onlyOwner
	updateClaimWait	External	✓	onlyOwner
	getClaimWait	External		-
	getTotalDividendsDistributed	External		-
	isExcludedFromFees	Public		-
	isExcludedFromMaxTx	Public		-
	isExcludedFromMaxWallet	Public		-
	withdrawableDividendOf	Public		-
	dividendTokenBalanceOf	Public		-
	excludeFromDividends	External	✓	onlyOwner
	getAccountDividendsInfo	External		-
	getAccountDividendsInfoAtIndex	External		-
	processDividendTracker	External	✓	-
	claim	External	✓	-
	getLastProcessedIndex	External		-
	getNumberOfDividendTokenHolders	External		-
	changeSwapTokensAtAmount	External	✓	onlyOwner
	extraFeeOnSell	Internal		
	_transfer	Internal	✓	
	swapAndLiquify	Private	✓	
	swapTokensForEth	Private	✓	
	manualSwapTokensForEth	Public	✓	onlyOwner
	swapTokensForETH	Private	✓	
	addLiquidity	Private	✓	
	swapAndSendDividends	Private	✓	
	withdrawStuckBNB	External	✓	onlyOwner
	removeStuckToken	External	✓	onlyOwner
ETHSHIBDividendTracker	Implementation	Ownable, DividendPayingToken		
	<Constructor>	Public	✓	DividendPayingToken
	_transfer	Internal		
	withdrawDividend	Public		-
	excludeFromDividends	External	✓	onlyOwner

	updateClaimWait	External	✓	onlyOwner
	getLastProcessedIndex	External		-
	getNumberOfTokenHolders	External		-
	getAccount	Public		-
	getAccountAtIndex	Public		-
	canAutoClaim	Private		
	setBalance	External	✓	onlyOwner
	process	Public	✓	-
	processAccount	Public	✓	onlyOwner

Contract Flow



Domain Info

Domain Name	ethshiba.io
Registry Domain ID	f317d4e1148f4dcb8544d434edb9c0ca-DONUTS
Creation Date	2022-03-04T02:22:54Z
Updated Date	2022-03-20T06:53:40Z
Registry Expiry Date	2023-03-04T02:22:54Z
Registrar WHOIS Server	whois.advancedregistrar.com
Registrar URL	http://www.netearthone.com
Registrar	NetEarth One Inc. dba NetEarth
Registrar IANA ID	1005

The domain has been created 3 months before the creation of the audit. It will expire in 9 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 stopping transactions, manipulating fees, transferring funds to the team’s wallet and blacklisting addresses. The contract can be converted into a honeypot and prevent users from selling if the owner abuses the admin functions. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats.

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