Cyberscope

Audit Report

SLAMDUNK INU

July 2022

Type	BEP20
Network	BSC
Address	0xdc2c13885AcF97b748823cdC61BE5B2DF0BdfFF0
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	SLAMDUNKINU
Compiler Version	v0.8.7+commit.e28d00a7
Optimization	200 runs
Licence	MIT
Explorer	https://bscscan.com/token/0xdc2c13885AcF97b748823cdC61BE5B2DF0BdfFF0
Symbol	SDI
Decimals	9
Total Supply	1,000,000
Domain	

Source Files

Filename	**SHA256**
contract.sol	14795712c05c17dc8da63860936458cd764834c87af51339afaa67f387493be2

Audit Updates

Initial Audit	9th July 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#L545

Description

The contract owner has the authority to stop tha sales for all users excluding the owner. The owner may take advantage of it by setting the `totalFee` to zero and transferring tokens to the contract address. As a result, the swapBack method will revert since the totalFee is used as denominator.

```
function swapBack() internal swapping {
    uint256 dynamicLiquidityFee = isOverLiquified(targetLiquidity,
targetLiquidityDenominator) ? 0 : liquidityFee;
    uint256 amountToLiquify =
swapThreshold.mul(dynamicLiquidityFee).div(totalFee).div(2);
```

Recommendation

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

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#L680

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 `setFees` function with a high percentage value.

```
function setFees(uint256 _liquidityFee, uint256 _buybackFee, uint256
_reflectionFee, uint256 _marketingFee, uint256 _feeDenominator) external
authorized {
    liquidityFee = _liquidityFee;
    buybackFee = _buybackFee;
    reflectionFee = _reflectionFee;
    marketingFee = _marketingFee;
    totalFee =
_liquidityFee.add(_buybackFee).add(_reflectionFee).add(_marketingFee);
    feeDenominator = _feeDenominator;
}
```

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#L704

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 `manualSend` method.

```
function manualSend() external authorized {
    uint256 contractETHBalance = address(this).balance;
    payable(marketingFeeReceiver).transfer(contractETHBalance);
}
```

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

Severity	Code	Description
●	L01	Public Function could be Declared External
●	L02	State Variables could be Declared Constant
●	L04	Conformance to Solidity Naming Conventions
●	L05	Unused State Variable
●	L07	Missing Events Arithmetic
●	L09	Dead Code Elimination

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L99,106,127,592,717

Description

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

```
getUnpaidEarnings
triggerManualBuyback
transferOwnership
unauthorize
authorize
```

Recommendation

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

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L207,220,366,364,365,367,373

Description

Constant state variables should be declared constant to save gas.

```
_totalSupply
ZERO
WBNB
DOGE
DEAD
dividendsPerShareAccuracyFactor
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L142,245,198,206,207,628,638,680,689,694,699,709,364,365,366,367,369,370,371,373,374,377,378

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.

```
_allowances
_balances
_maxTxAmount
_totalSupply
_decimals
_symbol
_name
ZERO
DEAD
...
```

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#L364

Description

There are segments that contain unused state variables.

```
DOGE
```

Recommendation

Remove unused state variables.

L07 - Missing Events Arithmetic

Criticality	minor
Location	contract.sol#L245,628,638,642,657,680,694,699

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.

```
targetLiquidity = _target
swapThreshold = _amount
liquidityFee = _liquidityFee
_maxTxAmount = amount
buybackMultiplierNumerator = numerator
deadBlocks = _deadBlocks
autoBuybackCap = _cap
minPeriod = _minPeriod
```

Recommendation

Emit an event for critical parameter changes.

L09 - Dead Code Elimination

Criticality	minor
Location	contract.sol#L507

Description

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

```
checkTxLimit
```

Recommendation

Remove unused functions.

Contract Functions

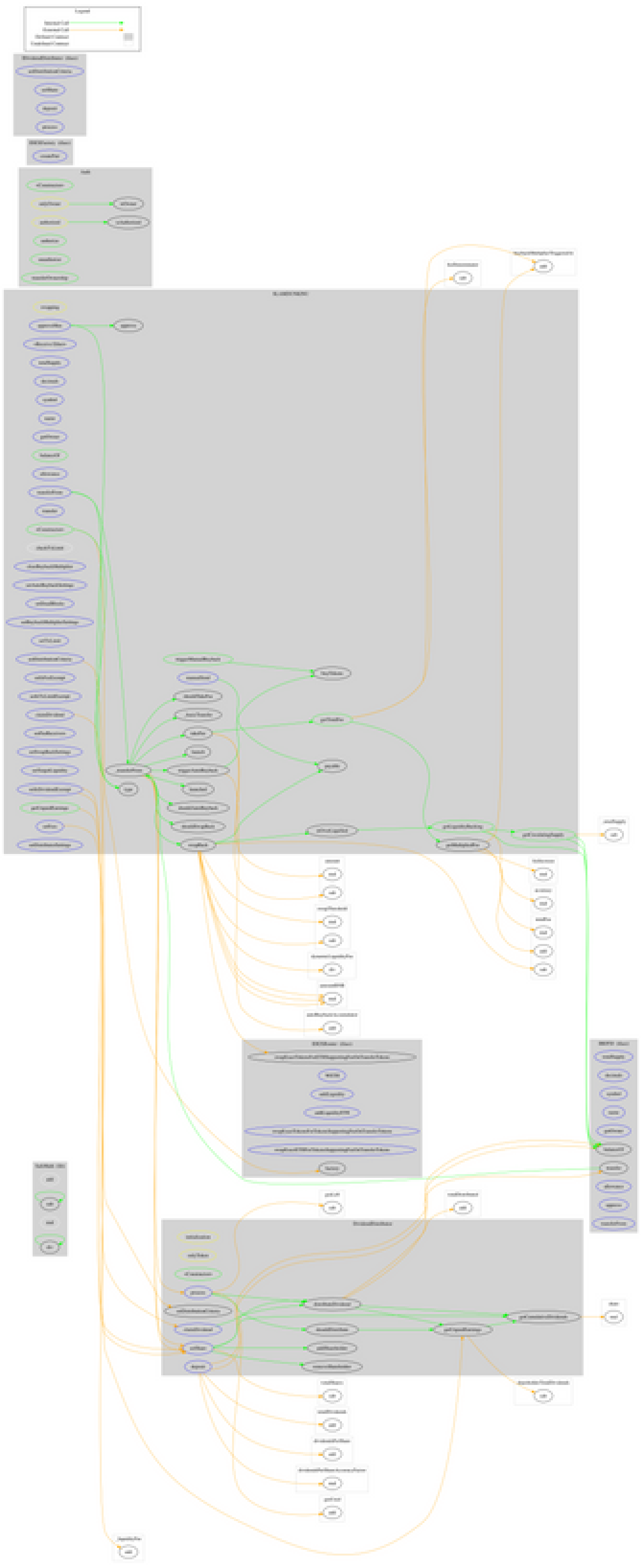
Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
SafeMath	Library			
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
IBEP20	Interface			
	totalSupply	External		-
	decimals	External		-
	symbol	External		-
	name	External		-
	getOwner	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
Auth	Implementation			
	<Constructor>	Public	✓	-
	authorize	Public	✓	onlyOwner
	unauthorize	Public	✓	onlyOwner
	isOwner	Public		-
	isAuthorized	Public		-
	transferOwnership	Public	✓	onlyOwner
IDEXFactory	Interface			

	createPair	External	✓	-
IDEXRouter	Interface			
	factory	External		-
	WETH	External		-
	addLiquidity	External	✓	-
	addLiquidityETH	External	Payable	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
IDividendDistributor	Interface			
	setDistributionCriteria	External	✓	-
	setShare	External	✓	-
	deposit	External	Payable	-
	process	External	✓	-
DividendDistributor	Implementation	IDividendDistributor		
	<Constructor>	Public	✓	-
	setDistributionCriteria	External	✓	onlyToken
	setShare	External	✓	onlyToken
	deposit	External	Payable	onlyToken
	process	External	✓	onlyToken
	shouldDistribute	Internal		
	distributeDividend	Internal	✓	
	claimDividend	External	✓	onlyToken
	getUnpaidEarnings	Public		-
	getCumulativeDividends	Internal		
	addShareholder	Internal	✓	
	removeShareholder	Internal	✓	
SLAMDUNKIN	Implementation	IBEP20,		

U		Auth		
	<Constructor>	Public	✓	Auth
	<Receive Ether>	External	Payable	-
	totalSupply	External		-
	decimals	External		-
	symbol	External		-
	name	External		-
	getOwner	External		-
	balanceOf	Public		-
	allowance	External		-
	approve	Public	✓	-
	approveMax	External	✓	-
	transfer	External	✓	-
	transferFrom	External	✓	-
	_transferFrom	Internal	✓	
	_basicTransfer	Internal	✓	
	checkTxLimit	Internal		
	shouldTakeFee	Internal		
	getTotalFee	Public		-
	getMultipliedFee	Public		-
	takeFee	Internal	✓	
	shouldSwapBack	Internal		
	swapBack	Internal	✓	swapping
	shouldAutoBuyback	Internal		
	triggerManualBuyback	Public	✓	onlyOwner
	clearBuybackMultiplier	External	✓	authorized
	triggerAutoBuyback	Internal	✓	
	buyTokens	Internal	✓	swapping
	setAutoBuybackSettings	External	✓	authorized
	setDeadBlocks	External	✓	authorized
	setBuybackMultiplierSettings	External	✓	authorized
	launched	Internal		
	launch	Internal	✓	
	setTxLimit	External	✓	authorized
	setIsDividendExempt	External	✓	authorized

	setIsFeeExempt	External	✓	authorized
	setIsTxLimitExempt	External	✓	authorized
	setFees	External	✓	authorized
	setFeeReceivers	External	✓	authorized
	setSwapBackSettings	External	✓	authorized
	setTargetLiquidity	External	✓	authorized
	manualSend	External	✓	authorized
	setDistributionCriteria	External	✓	authorized
	claimDividend	External	✓	-
	getUnpaidEarnings	Public		-
	setDistributorSettings	External	✓	authorized
	getCirculatingSupply	Public		-
	getLiquidityBacking	Public		-
	isOverLiquified	Public		-

Contract Flow



Summary

There are some functions that can be abused by the owner like stopping transactions, manipulating fees and transferring funds to the team's wallet. 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