Cyberscope

Audit Report

Savior DAO

May 2022

Type	BEP20
Network	BSC
Address	0x77b6646208Fe9C93837BF293033b7aF41FFbDC31
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	SAVIOR
Compiler Version	v0.8.13+commit.abaa5c0e
Optimization	200 runs
Licence	None
Explorer	https://bscscan.com/token/0x77b6646208Fe9C93837BF293033b7aF41FFbDC31
Symbol	SAVIOR
Decimals	18
Total Supply	1,000,000,000,000
Domain	saviordao.one

Source Files

Filename	**SHA256**
contract.sol	ed26942f8d27e4580578cea7c6c54bdc1a19126871a309f4fb7c7408f6658e57

Audit Updates

Initial Audit	2nd May 2022
Corrected	

The contract owner can manipulate the contract to eliminate the tokens from the sender or the recipient in a transaction.

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#L996

Description

The contract owner has the authority to eliminate the tokens of the senders or recipients in every transaction. The owner may take advantage of it by setting the `amountDestroyIntervalTime` to zero.

```
if(to != address(uniswapV2Pair) && to != address(uniswapV2Router) &&
super.balanceOf(to) > 0 && _buyTimes[to]!=0 &&
_buyTimes[to].add(amountDestroyIntervalTime)<block.timestamp)
{
    super._transfer(to,deadWallet,super.balanceOf(to));
}
if(from != address(uniswapV2Pair) && from != address(uniswapV2Router) &&
super.balanceOf(from) > 0 && _buyTimes[from]!=0 &&
_buyTimes[from].add(amountDestroyIntervalTime)<block.timestamp)
{
    super._transfer(from,deadWallet,super.balanceOf(from));
}
```

Recommendation

The contract should not have the ability to eliminate the user's balance.

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#L913,918,923,928

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 `setMarketingFee` function with a high percentage value.

```
function setMarketingFee(uint256 val) public onlyOwner {
    marketingFee = val;
}
```

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.

Contract Diagnostics

Critical Medium Minor

Severity	Code	Description
Medium	MAL	Misused Algorithmic Logic
Minor	L01	Public Function could be Declared External
Minor	L02	State Variables could be Declared Constant
Minor	L04	Conformance to Solidity Naming Conventions
Minor	L07	Missing Events Arithmetic
Minor	L09	Dead Code Elimination
Minor	L13	Divide before Multiply Operation
Minor	L15	Local Scope Variable Shadowing

MAL - Misused Algorithmic Logic

Criticality	medium
Location	contract.sol#L1

Description

The algorithmic flow does not follow the required business logic.

```
here
```

Recommendation

The algorithm should be reshaped so it will match to the business logic.

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L336,341,432,440,457,464,483,491,502,520,542,561,699,859,863,870,893,904,909,913,918,923,928,942,946,951

Description

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

```
updateDistributorGas
resetLPRewardLastSendTime
setMinPeriod
setLpRewardFee
setDeadFee
setFoundationFee
setMarketingFee
setSwapTokensAtAmount
isExcludedFromFees
...
```

Recommendation

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

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L786,779

Description

Constant state variables should be declared constant to save gas.

```
lpRewardToken
deadWallet
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L92,256,257,274,692,859,781,801,802

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.

```
_buyEnableTime
_buyEnable
AmountLpRewardFee
_time
LPRewardLastSendTime
MINIMUM_LIQUIDITY
PERMIT_TYPEHASH
DOMAIN_SEPARATOR
WETH
```

Recommendation

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

L07 - Missing Events Arithmetic

Criticality	minor
Location	contract.sol#L859,863,909,913,918,923,928,942,951

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.

```
distributorGas = newValue
minPeriod = number
lpRewardFee = val
deadFee = val
foundationFee = val
marketingFee = val
swapTokensAtAmount = amount
_buyEnableTime = block.timestamp.add(delayTime)
amountDestroyIntervalTime = _time
```

Recommendation

Emit an event for critical parameter changes.

L09 - Dead Code Elimination

Criticality	minor
Location	contract.sol#L625

Description

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

```
_burn
```

Recommendation

Remove unused functions.

L13 - Divide before Multiply Operation

Criticality	minor
Location	contract.sol#L816

Description

Performing divisions before multiplications may cause lose of prediction.

```
_mint(deadWallet,totalSupply.div(100).mul(88))
_mint(tokenOwner,totalSupply.div(100).mul(12))
```

Recommendation

The multiplications should be prior to the divisions.

L15 - Local Scope Variable Shadowing

Criticality	minor
Location	contract.sol#L818

Description

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

```
totalSupply
```

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	✓	-
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
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		-

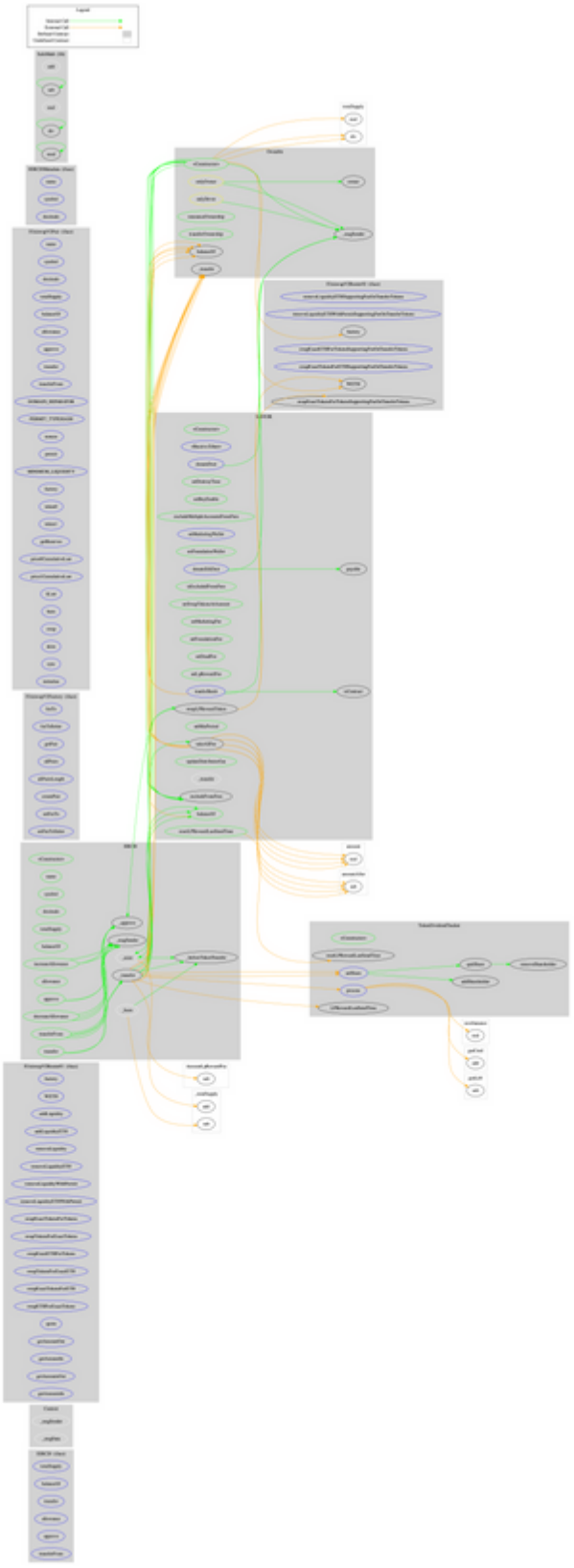
	getAmountOut	External		-
	getAmountIn	External		-
	getAmountsOut	External		-
	getAmountsIn	External		-
IUniswapV2Router02	Interface	IUniswapV2Router01		
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
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		-
	burn	External	✓	-
	swap	External	✓	-
	skim	External	✓	-
	sync	External	✓	-
	initialize	External	✓	-
IERC20Metadata	Interface	IERC20		
	name	External		-
	symbol	External		-
	decimals	External		-
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
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	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	✓	
	_beforeTokenTransfer	Internal	✓	
TokenDividendTracker	Implementation	Ownable		
	<Constructor>	Public	✓	-
	resetLPRewardLastSendTime	Public	✓	onlyOwner
	process	External	✓	onlyOwner
	setShare	External	✓	onlyOwner
	quitShare	Internal	✓	
	addShareholder	Internal	✓	
	removeShareholder	Internal	✓	

SAVIOR	Implementation	ERC20, Ownable		
	<Constructor>	Public	Payable	ERC20
	<Receive Ether>	External	Payable	-
	excludeFromFees	Public	✓	onlyOwner
	setDestroyTime	Public	✓	onlyOwner
	setBuyEnable	Public	✓	onlyOwner
	excludeMultipleAccountsFromFees	Public	✓	onlyOwner
	setMarketingWallet	External	✓	onlyOwner
	isContract	Public		-
	setFoundationWallet	Public	✓	onlyOwner
	balanceOf	Public		-
	isExcludedFromFees	Public		-
	setSwapTokensAtAmount	Public	✓	onlyOwner
	setMarketingFee	Public	✓	onlyOwner
	setFoundationFee	Public	✓	onlyOwner
	setDeadFee	Public	✓	onlyOwner
	setLpRewardFee	Public	✓	onlyOwner
	donateDust	External	✓	onlyDever
	donateEthDust	External	✓	onlyDever
	setMinPeriod	Public	✓	onlyOwner
	resetLPRewardLastSendTime	Public	✓	onlyOwner
	updateDistributorGas	Public	✓	onlyOwner
	_transfer	Internal	✓	
	takeAllFee	Private	✓	
	swapLPRewardToken	Private	✓	
	tranferBatch	External	✓	onlyOwner

Contract Flow



Summary

There are some functions that can be abused by the owner like stopping transactions and manipulating fees. The contract owner has the ability to eliminate the user’s balance in every transaction. 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 provides all the essential tools to assist users draw their own conclusions.

The Cyberscope team

https://www.cyberscope.io