



Cyberscope

Audit Report

BSCSTAR

April 2022

Type BEP20

Network BSC

Address 0x782f368d2244a58f3b49e57ee43797141ddc1220

Audited by © cyberscope

Table of Contents

Table of Contents	1
Contract Review	3
Audit Updates	3
Contract Analysis	4
ULTW - Unlimited Liquidity to Team Wallet	4
Description	5
Recommendation	5
Contract Diagnostics	6
FSA - Fixed Swap Address	7
Description	7
Recommendation	7
CO - Code Optimization	8
Description	8
Recommendation	8
L01 - Public Function could be Declared External	9
Description	9
Recommendation	9
L02 - State Variables could be Declared Constant	10
Description	10
Recommendation	10
L04 - Conformance to Solidity Naming Conventions	11
Description	11
Recommendation	11
L05 - Unused State Variable	12
Description	12
Recommendation	12

L07 - Missing Events Arithmetic	13
Description	13
Recommendation	13
L09 - Dead Code Elimination	14
Description	14
Recommendation	14
L13 - Divide before Multiply Operation	15
Description	15
Recommendation	15
Contract Functions	16
Contract Flow	22
Domain Info	23
Summary	24
Disclaimer	25
About Cyberscope	26

Contract Review

Contract Name	BCSSTAR
Compiler Version	v0.8.13+commit.abaa5c0e
Optimization	200 runs
Licence	MIT
Explorer	https://bscscan.com/token/0x782f368d2244a58f3b49e57ee43797141ddc1220
Symbol	BCSS
Decimals	18
Total Supply	100,000,000
Domain	bscstar.co

Source Files

Filename	SHA256
contract.sol	63ae5325c066812ba3fbbe3dc7e4abd6ca837eb38d8b5877c32a6edb4534e89d

Audit Updates

Initial Audit	2nd April 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

ULTW - Unlimited Liquidity to Team Wallet

Criticality	minor
Location	contract.sol#L1366, 1373, 1378, 1383

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 `emergencyTokenWithdraw`, `emergencyLPWithdraw`, `emergencyBNBWithdraw`, or `emergencyUSDTWithdraw` functions.

```
function emergencyTokenWithdraw(uint256 _amount) public onlyOwner {
    require(_amount <= balanceOf(address(this)), 'not enough token');
    _tOwned[address(this)] = _tOwned[address(this)].sub(_amount);
    _tOwned[address(msg.sender)] =
    _tOwned[address(msg.sender)].add(_amount);
    emit Transfer(address(this), msg.sender, _amount);
}
```

Recommendation

The contract could embody a check for the maximum amount of funds that can be swapped. Since a huge amount may violate 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
●	FSA	Fixed Swap Address
●	CO	Code Optimization
●	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
●	L13	Divide before Multiply Operation

FSA - Fixed Swap Address

Criticality	minor
Location	contract.sol#L924

Description

The swap address is assigned once in the constructor and it can not be changed. The decentralized swaps sometimes create a new swap version or abandon the current. A contract that cannot change the swap address may not be able to catch-up the upgrade.

```
IUniswapV2Router02 _uniswapV2Router = IUniswapV2Router02(  
    0x10ED43C718714eb63d5aA57B78B54704E256024E  
);  
  
    // Create a uniswap pair for this new token  
    uniswapV2Pair =  
    IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),  
    USDTAddress);  
  
    _approve(address(this), address(_uniswapV2Router), 2 ** 256 - 1);  
    IERC20(USDTAddress).safeApprove(address(_uniswapV2Router), 2 ** 256 -  
1);  
  
    // set the rest of the contract variables  
    uniswapV2Router = _uniswapV2Router;
```

Recommendation

It could be better to allow the swap address mutation in case of future swap updates.

CO - Code Optimization

Criticality

minor

Location

contract.sol#L901,1110,

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.

```
mapping(address => bool) private _marketList;
```

```
require( !_marketList[from], "ERC20: market is not enabled");  
require( !_marketList[to], "ERC20: market is not enabled");
```

Recommendation

Remove the `_marketList` definition and usage from the code so the runtime will be more performant.

L01 - Public Function could be Declared External

Criticality

minor

Location

contract.sol#L106,115,120,947,951,955,959,967,976,985 and 15 more

Description

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

```
setSwapAndLiquifyEnabled
emergencyUSDTWithdraw
emergencyBNBWithdraw
emergencyLPWithdraw
emergencyTokenWithdraw
updateReceiver
includeInFee
includeReward
excludeReward
...
```

Recommendation

Use the external attribute for functions never called from the contract

L02 - State Variables could be Declared Constant

Criticality

minor

Location

contract.sol#L864,862,863,878,96,95

Description

Constant state variables should be declared constant to save gas.

```
_previousOwner  
_lockTime  
_tTotal  
_symbol  
_name  
_decimals
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality

minor

Location

contract.sol#L560,562,593,639,1064,1366,1373,1383,1388,1392 and 11 more

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.

```
LPFeeDividends
_investerFee
_taxFee
_LPFee
_burnFee
USDTAddress
_enabled
_dividendsMinAmount
LPFee
...
```

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#L95,96

Description

There are segments that contain unused state variables.

```
_lockTime  
_previousOwner
```

Recommendation

Remove unused state variables.

L07 - Missing Events Arithmetic

Criticality

minor

Location

contract.sol#L1392,1396,1400,1404,1409,1414,1419,1424

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.

```
dividendsMinAmount = _dividendsMinAmount  
_inviterFee = inviterFee  
_taxFee = taxFee  
_LPFee = LPFee  
_burnFee = burnFee  
minPeriod = _minPeriod  
distributorGas = _distributorGas  
minTokenNumberToSell = _minTokenNumberToSell
```

Recommendation

Emit an event for critical parameter changes.

L09 - Dead Code Elimination

Criticality	minor
Location	contract.sol#L321,350,364,295,465,456,424

Description

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

```
safeTransferFrom  
safeIncreaseAllowance  
safeDecreaseAllowance  
sendValue  
functionCallWithValue  
functionCall
```

Recommendation

Remove unused functions.

L13 - Divide before Multiply Operation

Criticality

minor

Location

contract.sol#L1284,1432

Description

Performing divisions before multiplications may cause lose of prediction.

```
Transfer(sender,recipient,tAmount.div(10000).mul(recipientRate))
_tOwned[recipient] = _tOwned[recipient].add(tAmount.div(10000).mul(recipientRate))
_takeTaxFee(sender,tAmount.div(10000).mul(_taxFee))
_takeLPFee(sender,tAmount.div(10000).mul(_LPFee))
_takeburnFee(sender,tAmount.div(10000).mul(_burnFee))
curTAmount = tAmount.div(10000).mul(rate)
Transfer(sender,address(this),tAmount.div(10000).mul(_inviterFee.sub(accurRate)))
_tOwned[address(this)] =
_tOwned[address(this)].add(tAmount.div(10000).mul(_inviterFee.sub(accurRate)))
Transfer(sender,address(this),tAmount.div(10000).mul(_inviterFee))
...
```

Recommendation

The multiplications should be prior to the divisions.

Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
Ownable	Implementation			
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
SafeMath	Library			
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
Address	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	

	functionCallWithValue	Internal	✓	
	_functionCallWithValue	Private	✓	
SafeERC20	Library			
	safeTransfer	Internal	✓	
	safeTransferFrom	Internal	✓	
	safeApprove	Internal	✓	
	safeIncreaseAllowance	Internal	✓	
	safeDecreaseAllowance	Internal	✓	
	_callOptionalReturn	Private	✓	
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		-

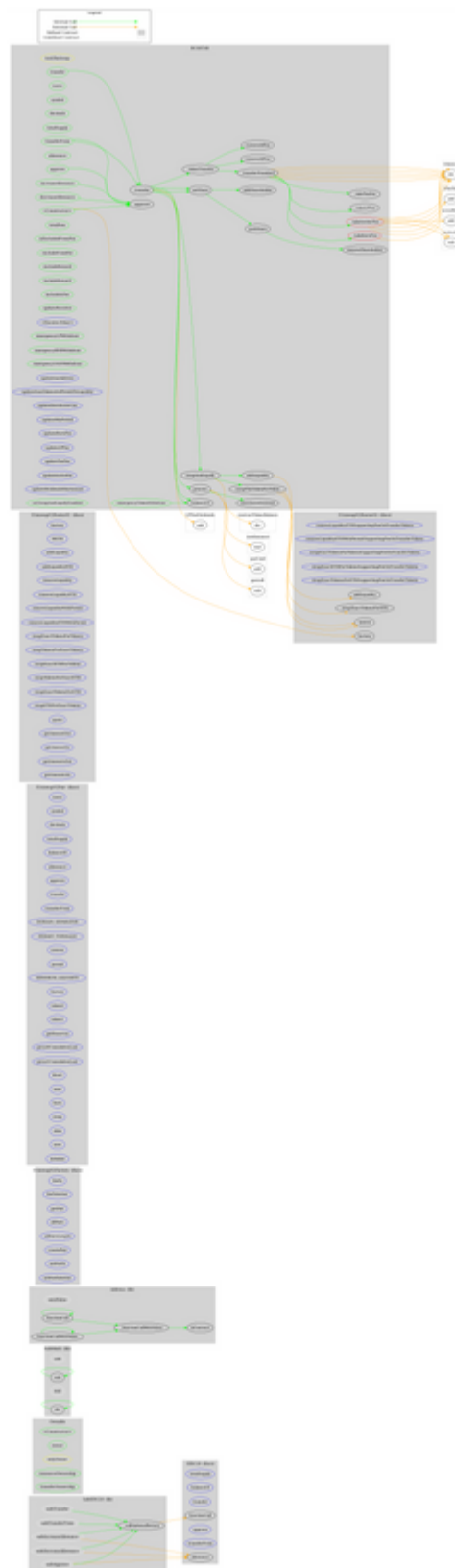
	nonces	External		-
	permit	External	✓	-
	MINIMUM_LIQUIDITY	External		-
	factory	External		-
	token0	External		-
	token1	External		-
	getReserves	External		-
	price0CumulativeLast	External		-
	price1CumulativeLast	External		-
	kLast	External		-
	mint	External	✓	-
	burn	External	✓	-
	swap	External	✓	-
	skim	External	✓	-
	sync	External	✓	-
	initialize	External	✓	-
IUniswapV2Router01	Interface			
	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	✓	-
BCSSTAR	Implementation	IERC20, Ownable		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	totalFees	Public		-
	isExcludedFromFee	Public		-
	excludeFromFee	Public	✓	onlyOwner
	excludeReward	Public	✓	onlyOwner
	includeReward	Public	✓	onlyOwner
	includeInFee	Public	✓	onlyOwner

	updateReceiver	Public	✓	onlyOwner
	<Receive Ether>	External	Payable	-
	removeAllFee	Private	✓	
	restoreAllFee	Private	✓	
	_approve	Private	✓	
	_transfer	Private	✓	
	swapAndLiquify	Private	✓	lockTheSwap
	process	Private	✓	
	distributeDividend	Internal	✓	
	setShare	Private	✓	
	addShareholder	Internal	✓	
	quitShare	Private	✓	
	removeShareholder	Internal	✓	
	_tokenTransfer	Private	✓	
	_takeburnFee	Private	✓	
	_takeLPFee	Private	✓	
	_takeTaxFee	Private	✓	
	_takeInviterFee	Private	✓	
	swapThisTokenForToken	Private	✓	
	addLiquidity	Private	✓	
	emergencyTokenWithdraw	Public	✓	onlyOwner
	emergencyLPWithdraw	Public	✓	onlyOwner
	emergencyBNBWithdraw	Public	✓	onlyOwner
	emergencyUSDTWithdraw	Public	✓	onlyOwner
	updateTaxAddress	External	✓	onlyOwner
	updateNumTokensSellToAddToLiquidity	External	✓	onlyOwner
	updateDistributorGas	External	✓	onlyOwner
	updateMinPeriod	External	✓	onlyOwner
	updateBurnFee	External	✓	onlyOwner
	updateLPFee	External	✓	onlyOwner
	updateTaxFee	External	✓	onlyOwner
	updateInviteFee	External	✓	onlyOwner
	updateDividendsMinAmount	External	✓	onlyOwner

	setSwapAndLiquifyEnabled	Public	✓	onlyOwner
	_transferStandard	Private	✓	

Contract Flow



Domain Info

Domain Name	bscstar.co
Registry Domain ID	D2D9578FC016045FBA708FB338914AC77-GDREG
Creation Date	2022-03-29T08:48:22Z
Updated Date	2022-04-02T02:20:09Z
Registry Expiry Date	2023-03-29T08:48:22Z
Registrar WHOIS Server	
Registrar URL	www.name.com
Registrar	Name.com, Inc.
Registrar IANA ID	625

The domain has been created 4 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

BSCSTAR is an interesting project that has a friendly and growing community. The Smart Contract analysis reported no compiler error or critical issues. The contract Owner can access some admin functions that can not be used in a malicious way to disturb the users' transactions. There is also a limit of max 25% 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 provides all the essential tools to assist users draw their own conclusions.



The Cyberscope team

<https://www.cyberscope.io>