Cyberscope

Audit Report

SPORTY

August 2022

Type	BEP20
Network	BSC
Address	0xe672237cC26FcfDFb593e30E7fEff450646c8A3f
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	Sporty
Compiler Version	v0.8.15+commit.e14f2714
Optimization	200 runs
Licence	None
Explorer	https://bscscan.com/token/0xe672237cc26fcfdfb593e30e7feff450646c8a3f
Symbol	SPƆRTY
Decimals	5
Total Supply	1,000,000,000
Domain	sporty.game

Source Files

Filename	**SHA256**
contract.sol	e87cac03f9c35fa4ce301ae979644b1bda363598b6ddde1bf48ec23217170fc9

Audit Updates

Initial Audit	20th August 2022
Corrected	

Contract Analysis

Critical Medium Minor Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Passed
●	OCTD	Transfers Contract's Tokens	Passed
●	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

ULTW - Transfers Liquidity to Team Wallet

Criticality	minor
Location	contract.sol#L553, 560,435
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 `clearStuckToken` and `clearStuckBalance` methods.

```
function clearStuckToken(address tokenAddress, uint256 tokens) public onlyOwner
returns (bool) {
     if(tokens == 0){
            tokens = IBEP20(tokenAddress).balanceOf(address(this));
        }
        return IBEP20(tokenAddress).transfer(msg.sender, tokens);
    }

function clearStuckBalance(uint256 amountPercentage, address _ReceiverStuck)
external onlyOwner {
        uint256 amountBNB = address(this).balance;
        payable(_ReceiverStuck).transfer(amountBNB * amountPercentage / 100);
    }
```

Furthermore, The contract owner can take advantage of the method `triggerBuyback` and transfer tokens from the contract balance directly into the Staking address.

```
    function triggerBuyback(uint256 amount, bool triggerBuybackMultiplier)
external onlyOwner() {
        buyTokens(amount, Staking);
        if(triggerBuybackMultiplier){
            buybackMultiplierTriggeredAt = block.timestamp;
            emit BuybackMultiplierActive(buybackMultiplierLength);
        }}
```

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	Status
	ZD	Zero Division	Unresolved
	STC	Succeeded Transfer Check	Unresolved
	L01	Public Function could be Declared External	Unresolved
	L02	State Variables could be Declared Constant	Unresolved
	L04	Conformance to Solidity Naming Conventions	Unresolved
	L07	Missing Events Arithmetic	Unresolved
	L12	Using Variables before Declaration	Unresolved
	L14	Uninitialized Variables in Local Scope	Unresolved

ZD - Zero Division

Criticality	minor
Location	contract.sol#L382
Status	Unresolved

Description

The contract is using variables that may be set to zero as denominators. As a result, the transactions will revert.

```
        uint256 amountToLiquify =
swapThreshold.mul(dynamicLiquidityFee).div(totalBuyFee).div(2);
```

Recommendation

The contract should prevent totalBuyFee to be set to zero or should not allow to execute the corresponding statements.

STC - Succeeded Transfer Check

Criticality	minor
Location	contract.sol#L407
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.

```
            (bool success, /* bytes memory data */) =
payable(marketingFeeReceiver).call{value: amountBNBMarketing, gas: 30000}("");
```

Recommendation

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

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L154,158,170
Status	Unresolved

Description

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

```
authorize
unauthorize
transferOwnership
```

Recommendation

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

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L185,187,184,186,193
Status	Unresolved

Description

Constant state variables should be declared constant to save gas.

```
DEAD
Staking
WBNB
ZERO
_totalSupply
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L91,496,508,514,519,544,560,184,185,186,187,189,190,191,193,194,196,197,203,205,213
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.

```
WETH
_liquidityFee
_StakingFee
_marketingFee
_SportyInsurancePool
_feeDenominator
_totalSellFee
_autoLiquidityReceiver
_marketingFeeReceiver
...
```

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#L466,482,496,514,519,549
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.

```
buybackMultiplierNumerator = numerator
_maxTxAmount = amount
liquidityFee = _liquidityFee
swapThreshold = _amount
targetLiquidity = _target
launchedAt = launched_
```

Recommendation

Emit an event for critical parameter changes.

L12 - Using Variables before Declaration

Criticality	minor
Location	contract.sol#L428,459
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.

```
e
reason
```

Recommendation

The variables should be declared before any usage of them.

L14 - Uninitialized Variables in Local Scope

Criticality	minor
Location	contract.sol#L459,428
Status	Unresolved

Description

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

```
reason
e
```

Recommendation

All the local scoped variables should be initialized.

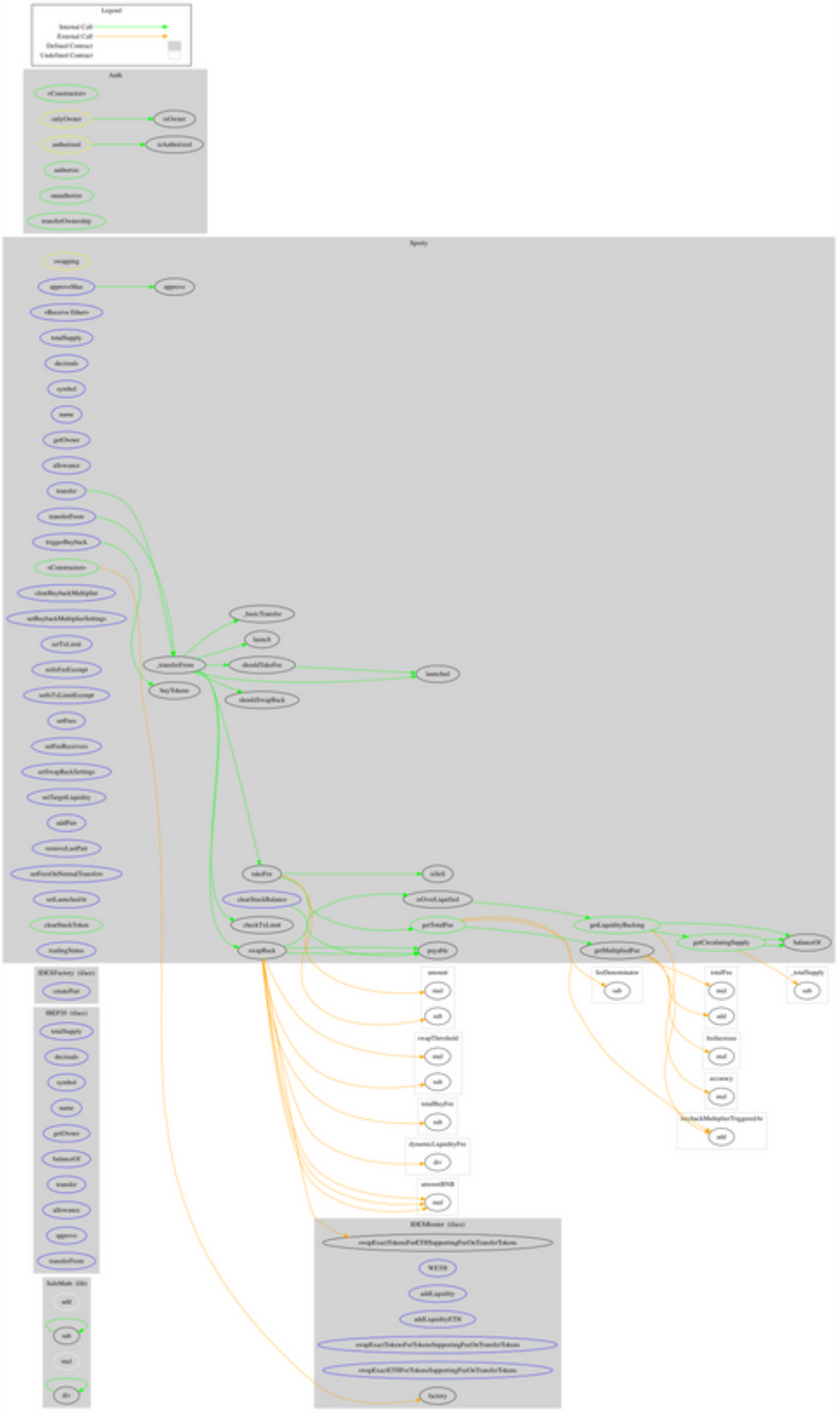
Contract Functions

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	✓	-
IDEXFactory	Interface			
	createPair	External	✓	-
IDEXRouter	Interface			
	factory	External		-
	WETH	External		-
	addLiquidity	External	✓	-
	addLiquidityETH	External	Payable	-

	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
Auth	Implementation			
	<Constructor>	Public	✓	-
	authorize	Public	✓	onlyOwner
	unauthorize	Public	✓	onlyOwner
	isOwner	Public		-
	isAuthorized	Public		-
	transferOwnership	Public	✓	onlyOwner
Sporty	Implementation	IBEP20, 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	✓	

	isSell	Internal		
	shouldSwapBack	Internal		
	swapBack	Internal	✓	swapping
	triggerBuyback	External	✓	onlyOwner
	clearBuybackMultiplier	External	✓	onlyOwner
	buyTokens	Internal	✓	swapping
	setBuybackMultiplierSettings	External	✓	onlyOwner
	launched	Internal		
	launch	Internal	✓	
	setTxLimit	External	✓	onlyOwner
	setIsFeeExempt	External	✓	onlyOwner
	setIsTxLimitExempt	External	✓	onlyOwner
	setFees	External	✓	onlyOwner
	setFeeReceivers	External	✓	onlyOwner
	setSwapBackSettings	External	✓	onlyOwner
	setTargetLiquidity	External	✓	onlyOwner
	getCirculatingSupply	Public		-
	getLiquidityBacking	Public		-
	isOverLiquified	Public		-
	addPair	External	✓	onlyOwner
	removeLastPair	External	✓	onlyOwner
	setFeesOnNormalTransfers	External	✓	onlyOwner
	setLaunchedAt	External	✓	onlyOwner
	clearStuckToken	Public	✓	onlyOwner
	clearStuckBalance	External	✓	onlyOwner
	tradingStatus	External	✓	onlyOwner

Contract Flow



Summary

The Smart Contract analysis reported one minor severity issue. The contract owner has the authority to transfer funds to the team's wallet. Other than that, the contract owner can access some admin functions that can not be used in a malicious way to disturb the users' transactions. The contract fees can be changed up to 15%.

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