Cyberscope

Audit Report

PiSoccer

August 2022

Type	BEP20
Network	BSC
Address	0xe1365370C09b4a6e64D0a1C0f58cc097813C95Ce
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	PISO
Compiler Version	v0.8.7+commit.e28d00a7
Optimization	200 runs
Licence	None
Explorer	https://bscscan.com/token/0xe1365370C09b4a6e64D0a1C0f58cc097813C95Ce
Symbol	PISO
Decimals	9
Total Supply	3,000,000,000
Domain	pisoccer.world

Source Files

Filename	**SHA256**
contract.sol	6f566841105dc216bbbf58bcce6fb0eb9410343f603829fc02216d4f8cbb8827

Audit Updates

Initial Audit	1st August 2022
Corrected	3rd August 2022

Contract Analysis

● Critical ● Medium ● Minor ● Pass

Severity	Code	Description	Status
●	ST	Contract Owner is not able to stop or pause transactions	Semi-Resolved
●	OCTD	Contract Owner is not able to transfer tokens from specific address	Resolved
●	OTUT	Owner Transfer User's Tokens	
●	ELFM	Contract Owner is not able to increase fees more than a reasonable percent (25%)	Resolved
●	ULTW	Contract Owner is not able to increase the amount of liquidity taken by dev wallet more than a reasonable percent	Resolved
●	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	Resolved

ST - Stop Transactions

Criticality	critical
Location	contract.sol#L486
Status	Semi-Resolved

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 :

- tradingEnabled to false.
- maxSellLimit to zero (**HONEYPOT**).
- maxWalletLimit to zero.
- coolDownTime to maximum amount (**HONEYPOT**).

```
        if(!_isExcludedFromFee[from] && !_isExcludedFromFee[to]){
            require(tradingEnabled, "Trading not active");
        }

        if(from != pair && !_isExcludedFromFee[to] && !_isExcludedFromFee[from] && !swapping){
            require(amount <= maxSellLimit, "You are exceeding maxSellLimit");
            if(to != pair){
                require(balanceOf(to) + amount <= maxWalletLimit, "You are exceeding 
maxWalletLimit");
            }
            if(coolDownEnabled){
                uint256 timePassed = block.timestamp - _lastSell[from];
                require(timePassed >= coolDownTime, "Cooldown enabled");
                _lastSell[from] = block.timestamp;
            }
         }
```

Contract Reverts in Small Amounts

The contract has a hard limit of `10 *  10**decimals()`. The contract is able to stop the transactions if the user's balance is less than the hard limit. As a result, the transaction will underflow.

```
if(balanceOf(from) - amount <= 10 *  10**decimals()) amount -= (10 * 10**decimals() + amount -
balanceOf(from));
```

For instance

Balance	9
Amount	5

```
amount -= (10 * 10**decimals() + amount - balanceOf(from)) ->
amount -= 10 + 5 - 9 ->
amount -= 6 ->
5 -= 6
```

Recommendation

The contract could embody a check for not allowing setting the `maxSellLimit`, `maxWalletLimit` 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 contract could embody a check for not allowing setting the `coolDownTime` more than a reasonable amount.

The contract should not have a hard limit for the user's token.

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.

Updated 03 August 2022

The team has renounced ownership, as a result all the honeypot-related issues have been resolved. The finding with the contract underflow remains vulnerable since it is not affected by the contract’s permissions.

OCTD - Owner Contract Tokens Drain

Criticality	minor
Location	contract.sol#L720
Status	Resolved

Description

The contract owner has the authority to claim all the balance of the contract. The owner may take advantage of it by calling the `StopAirDrop` function.

```
function StopAirDrop(address _tokenAddr, address _to, uint _amount) public onlyOwner {
        IERC20(_tokenAddr).transfer(_to, _amount);
 }
```

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.

Updated 03 August 2022

The team has renounced ownership and resolved the issues.

ELFM - Exceed Limit Fees Manipulation

Criticality	critical
Location	contract.sol#L362,367
Status	Resolved

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 setTaxes, setSellTaxes function with a high percentage value.

```
    function setTaxes(uint256 _rfi, uint256 _marketing, uint256 _liquidity, uint256 _dev, uint256
_buyback) public onlyOwner {
        taxes = Taxes(_rfi,_marketing,_liquidity,_dev,_buyback);
         emit FeesChanged();
    }

    function setSellTaxes(uint256 _rfi, uint256 _marketing, uint256 _liquidity, uint256 _dev, uint256
_buyback) public onlyOwner {
        sellTaxes = Taxes(_rfi,_marketing,_liquidity,_dev,_buyback);
         emit FeesChanged();
    }
```

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.

Updated 03 August 2022

The team has renounced ownership and resolved the issues.

ULTW - Unlimited Liquidity to Team Wallet

Criticality	minor
Location	contract.sol#L702
Status	Resolved

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 `SendtoContract` method.

```
function SendtoContract(uint256 weiAmount) external onlyOwner{
        require(address(this).balance >= weiAmount, "insufficient BNB 
balance");
        payable(msg.sender).transfer(weiAmount);
    }
```

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.

Updated 03 August 2022

The team has renounced ownership and resolved the issues.

BC - Blacklisted Contracts

Criticality	critical
Location	contract.sol#L672,676
Status	Resolved

Description

The contract owner has the authority to stop contracts from transactions. The owner may take advantage of it by calling the updateIsBCheck, BCheck functions.

```
function updateIsBCheck(address account, bool state) external onlyOwner{
    _isBCheck[account] = state;
}

function BCheck(address[] memory accounts, bool state) external onlyOwner{
    for(uint256 i =0; i < accounts.length; i++){
        _isBCheck[accounts[i]] = state;

    }
}
```

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.

Updated 03 August 2022

The team has renounced ownership and resolved the issues.

Contract Diagnostics

● Critical ● Medium ● Minor

Severity	Code	Description	Status
●	ZD	Zero Division	Resolved
●	STC	Succeeded Transfer Check	
●	L01	Public Function could be Declared External	
●	L02	State Variables could be Declared Constant	
●	L04	Conformance to Solidity Naming Conventions	
●	L07	Missing Events Arithmetic	
●	L11	Unnecessary Boolean equality	
●	L13	Divide before Multiply Operation	

ZD - Zero Division

Criticality	critical
Location	contract.sol#L573
Status	Resolved

Description

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

```
function swapAndLiquify(uint256 contractBalance, Taxes memory temp) private
lockTheSwap{

        uint256 denominator = (temp.liquidity + temp.marketing + temp.dev +
temp.buyback) * 2;
        uint256 tokensToAddLiquidityWith = contractBalance * temp.liquidity /
denominator;
        uint256 toSwap = contractBalance - tokensToAddLiquidityWith;

        uint256 initialBalance = address(this).balance;

        swapTokensForBNB(toSwap);
```

Recommendation

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

Updated 03 August 2022

The team has renounced ownership, since the tax contains non-zero values, the denominator will never be able to produce a zero division issue .

STC - Succeeded Transfer Check

Criticality	minor
Location	contract.sol#L721

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.

```
function StopAirDrop(address _tokenAddr, address _to, uint _amount) public onlyOwner {
    IERC20(_tokenAddr).transfer(_to, _amount);
}
```

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#L246,59,279,358,716,287,274,354,704,264,297,63,293,363,238,259,255,349,235,345

Description

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

```
excludeFromFee
name
includeInFee
allowance
approve
symbol
setSellTaxes
isExcludedFromReward
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#L144

Description

Constant state variables should be declared constant to save gas.

```
_tTotal
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L363,358,159,152,698,716,154,141,642,81,156,672,182,646,158,704

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.

```
StartSale
_name
ExTax
valuesFromGetValues
_to
BCheck
_marketing
Charity
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#L637,664,683,688

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.

```
maxWalletLimit = amount * 10 ** decimals()
maxBuyLimit = maxBuy * 10 ** decimals()
swapTokensAtAmount = amount * 10 ** _decimals
coolDownTime = time * 1
```

Recommendation

Emit an event for critical parameter changes.

L11 - Unnecessary Boolean equality

Criticality	minor
Location	contract.sol#L309

Description

The comparison to boolean constants is redundant. Boolean constants can be used directly and do not need to be compared to true or false.

```
state == true && genesis_block == 0
```

Recommendation

Remove the equality to the boolean constant.

L13 - Divide before Multiply Operation

Criticality	minor
Location	contract.sol#L569

Description

Performing divisions before multiplications may cause lose of prediction.

```
unitBalance = deltaBalance / (denominator - temp.liquidity)
```

Recommendation

The multiplications should be prior to the divisions.

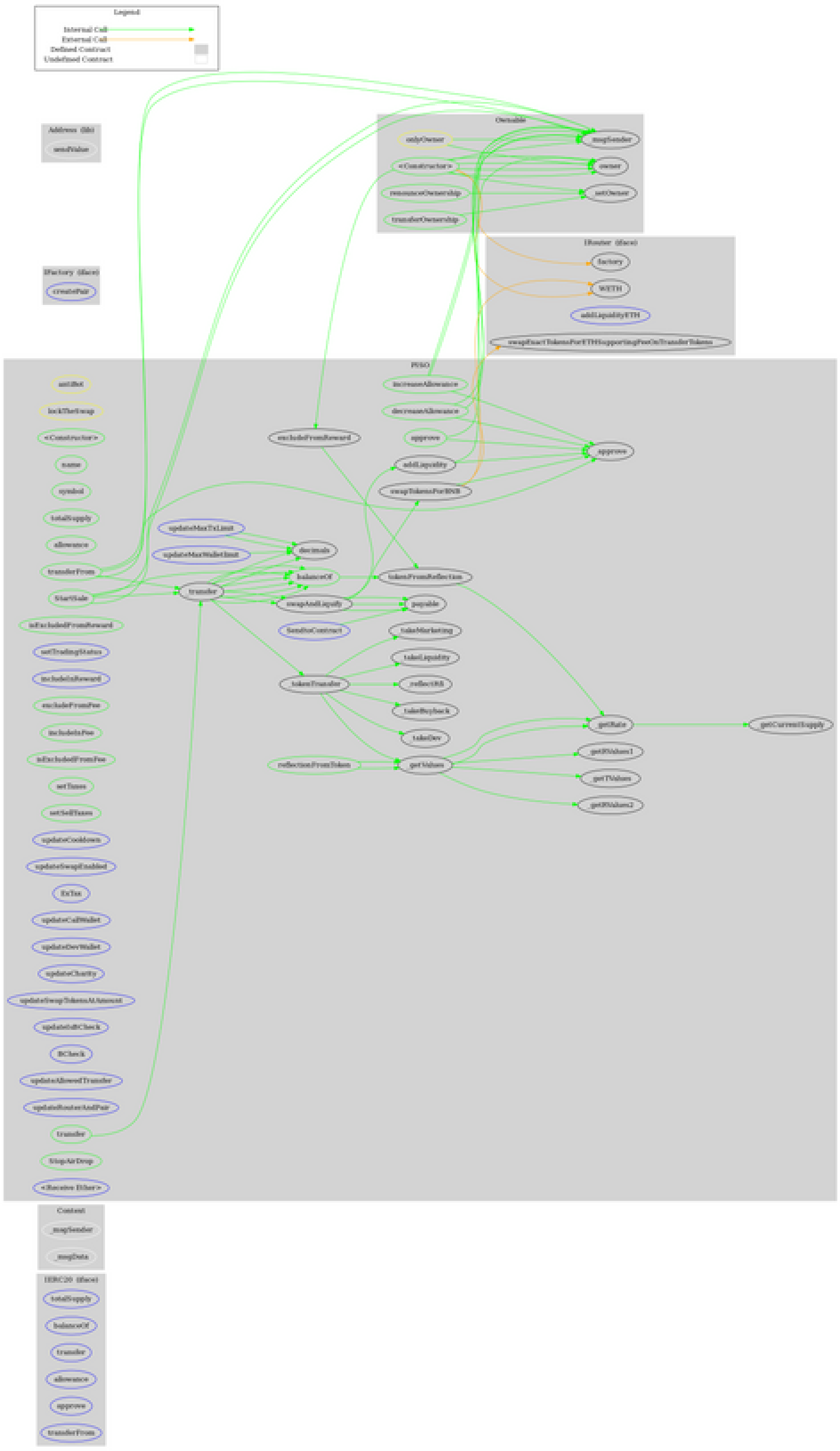
Contract Functions

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_setOwner	Private	✓	
IFactory	Interface			
	createPair	External	✓	-
IRouter	Interface			
	factory	External		-
	WETH	External		-
	addLiquidityETH	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-

Address	Library			
	sendValue	Internal	✓	
PISO	Implementation	Context, IERC20, Ownable		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	allowance	Public		-
	approve	Public	✓	antiBot
	transferFrom	Public	✓	antiBot
	increaseAllowance	Public	✓	antiBot
	decreaseAllowance	Public	✓	antiBot
	transfer	Public	✓	antiBot
	isExcludedFromReward	Public		-
	reflectionFromToken	Public		-
	setTradingStatus	External	✓	onlyOwner
	tokenFromReflection	Public		-
	excludeFromReward	Public	✓	onlyOwner
	includeInReward	External	✓	onlyOwner
	excludeFromFee	Public	✓	onlyOwner
	includeInFee	Public	✓	onlyOwner
	isExcludedFromFee	Public		-
	setTaxes	Public	✓	onlyOwner
	setSellTaxes	Public	✓	onlyOwner
	_reflectRfi	Private	✓	
	_takeLiquidity	Private	✓	
	_takeMarketing	Private	✓	
	_takeDev	Private	✓	
	_takeBuyback	Private	✓	
	_getValues	Private		
	_getTValues	Private		

	_getRValues1	Private		
	_getRValues2	Private		
	_getRate	Private		
	_getCurrentSupply	Private		
	_approve	Private	✓	
	_transfer	Private	✓	
	_tokenTransfer	Private	✓	
	swapAndLiquify	Private	✓	lockTheSwap
	addLiquidity	Private	✓	
	swapTokensForBNB	Private	✓	
	updateCooldown	External	✓	onlyOwner
	updateSwapEnabled	External	✓	onlyOwner
	ExTax	External	✓	onlyOwner
	updateCallWallet	External	✓	onlyOwner
	updateDevWallet	External	✓	onlyOwner
	updateCharity	External	✓	onlyOwner
	updateSwapTokensAtAmount	External	✓	onlyOwner
	updateIsBCheck	External	✓	onlyOwner
	BCheck	External	✓	onlyOwner
	updateAllowedTransfer	External	✓	onlyOwner
	updateMaxTxLimit	External	✓	onlyOwner
	updateMaxWalletlimit	External	✓	onlyOwner
	updateRouterAndPair	External	✓	onlyOwner
	SendtoContract	External	✓	onlyOwner
	StartSale	Public	✓	-
	StopAirDrop	Public	✓	onlyOwner
	<Receive Ether>	External	Payable	-

Contract Flow



Domain Info

Domain Name	pisoccer.world
Registry Domain ID	234e4d2a0da54233bb6d45e89564d120-DONUTS
Creation Date	2022-07-13T07:41:56Z
Updated Date	2022-07-19T08:38:37Z
Registry Expiry Date	2023-07-13T07:41:56Z
Registrar WHOIS Server	whois.namecheap.com
Registrar URL	https://www.namecheap.com/
Registrar	NameCheap, Inc.
Registrar IANA ID	1068

The domain has been created in 12 months before the creation of the audit.

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, transferring tokens to the team's wallet, manipulating fees, transferring funds to the team's wallet and massively blacklisting addresses.

The contract can be converted into a honeypot and prevent users from selling if the owner abuses the admin functions.

The contract stops sell transactions for the first three blocks.

The contract has a hard limit for User's balance of 10 tokens. The contract stops transactions if their balance is lower than the hard limit.

A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats.

Updated 03 August 2022

The team has renounced ownership and resolved the issues. The contract hard limit issue remains since it is not affected by the contract roles.

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