Cyberscope

Audit Report

Marco inu

May 2022

Type	BEP20
Network	BSC
Address	0x16be19fb70f902ecf39b54ad988ee427cb1b6848
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	MANU
Compiler Version	v0.6.12+commit.27d51765
Optimization	200 runs
Licence	Unlicense
Explorer	https://bscscan.com/token/0x16be19fb70f902ecf39b54ad988ee427cb1b6848
Symbol	MANU
Decimals	18
Total Supply	990,735,549,764
Domain	marcoinu.io

Source Files

Filename	**SHA256**
contract.sol	269277ffa00532dae2e9d0565ebbbf9bc6e3a13eb0b57bd23ec65e34ac0f8f5d

Audit Updates

Initial Audit	5th May 2022
Corrected	11th May 2022

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

Contract Diagnostics

Critical Medium Minor

Severity	Code	Description
●	BLC	Business Logic Concern
●	CO	Code Optimization
●	CR	Code Repetition
●	DSM	Data Structure Misuse
●	L01	Public Function could be Declared External
●	L02	State Variables could be Declared Constant
●	L04	Conformance to Solidity Naming Conventions
●	L09	Dead Code Elimination
●	L15	Local Scope Variable Shadowing

BLC - Business Logic Concern

Criticality	critical
Location	contract.sol#L1193

Description

The contract shares a proportion of the taxed amount to every user. Then, the same amount is used in order to receive taxes. Thus, the contract sends more than the expected taxed amount to addresses without updating the total supply. As a result, the address balance is diverse from the total supply.

```
if (amount_accounts != 0) {
    uint256 _sReward = rBurn.div(amount_accounts);
    uint256 _tReward = tLiquidity.div(amount_accounts);

    for (uint256 i = 0; i < _accounts.length; i ++) {
        address _temp = _accounts[i];
        if (_temp != sender && _temp != recipient) {
            _rOwned[_temp] = _rOwned[_temp].add(_sReward);
            _buyHistories[_temp].push(History(block.timestamp, _tReward,
"buy"));
        }
    }
}
```

The contract takes one fee amount twice.

```
_takeLiquidity(tLiquidity);
_reflectFee(rBurn, tLiquidity);
```

The contract contains inconsistency between the real taxed fees, the transferable amount after the taxes and the actual amount since it multiplies the taxed amount four times after the initial calculation.

```
function _getTValues(uint256 tAmount) private view returns (uint256, uint256) {
    uint256 tFee = calculateFee(tAmount);
    uint256 tTransferAmount = tAmount.sub(tFee.mul(4));
    return (tTransferAmount, tFee);
}
```

The contract is using a fee called “rBurn” in order to receive tazed in the marketing address.

```
_rOwned[marketing_address] = _rOwned[marketing_address].add(rBurn);
```

Recommendation

The team is advised to carefully check if the implementation follows the expected business logic.

CO - Code Optimization

Criticality	minor
Location	contract.sol#L1050

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.

The history status could use an enumerated number rather than a string

```
keccak256(bytes(_history.status)) == keccak256(bytes(buy))
```

Recommendation

Rewrite some code segments so the runtime will be more performant.

CR - Code Repetition

Criticality	minor
Location	contract.sol#L806,1044

Description

There are code segments that are repetitive in the contract. Those segments increase the code size of the contract unnecessarily. The code segment in getBalanceAbleSell() and in the transfer method are exactly the same.

```
string memory buy = "buy";
string memory sell = "sell";
uint256 allowedAmount = 0;
for(uint256 i = 0; i < _buyHistories[from].length; i++) {
    History memory _history = _buyHistories[from][i];
    if (keccak256(bytes(_history.status)) == keccak256(bytes(buy))) {
        allowedAmount = allowedAmount.add(_history.amount);
    }
    if (keccak256(bytes(_history.status)) == keccak256(bytes(sell))) {
        allowedAmount = allowedAmount.sub(_history.amount);
    }
}
```

Recommendation

Create an internal function that contains the code segment and remove it from all the sections.

DSM - Data Structure Misuse

Criticality	minor
Location	contract.sol#L1060

Description

The contract uses the valuable _accounts as an array. The business logic of the contract does not require to iterate this structure sequentially. Thus, unnecessary loops are produced that increase the required gas.

```
uint256 count = 0;
for (uint256 i = 0; i < _accounts.length; i ++) {
    if (_accounts[i] == to)
        count ++;
}
if (count == 0) {
    _accounts.push(to);
}
```

Recommendation

The contract could use a data structure that provides instant access. For instance, a Set would fit better to the business logic of the contract. This way the time complexity will be reduced from o(n) to o(1). The latest addition for the expression `from == _accounts[0]` could be keeped in a separate variable.

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L449,458,464,469,477,784,802,867,927,931,935,1016

Description

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

```
isExcludedFromFee
setSwapItAndLiquifyEnabled
includeInFee
excludeFromFee
excludeFromReward
getBalanceAbleSell
decimals
unlock
lock
...
```

Recommendation

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

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L716,714,715,739,740,728

Description

Constant state variables should be declared constant to save gas.

```
numTokensSellToAddToLiquidity
marketing_address
_timeLimit
_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#L502,522,523,540,562,802,935,998,718,739,740

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.

```
marketing_address
_timeLimit
_fee
_amount
_enabled
_address
WETH
MINIMUM_LIQUIDITY
PERMIT_TYPEHASH
...
```

Recommendation

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

L09 - Dead Code Elimination

Criticality	minor
Location	contract.sol#L375,335,345,360,370,282,309

Description

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

```
sendValue
isContract
functionCallWithValue
functionCall
_functionCallWithValue
```

Recommendation

Remove unused functions.

L15 - Local Scope Variable Shadowing

Criticality	minor
Location	contract.sol#L945

Description

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

```
_owner
```

Recommendation

The local variables should have different names from the upper scoped variables.

Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
IBEP20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
SafeMath	Library			
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
	mod	Internal		
	mod	Internal		
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Address	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	

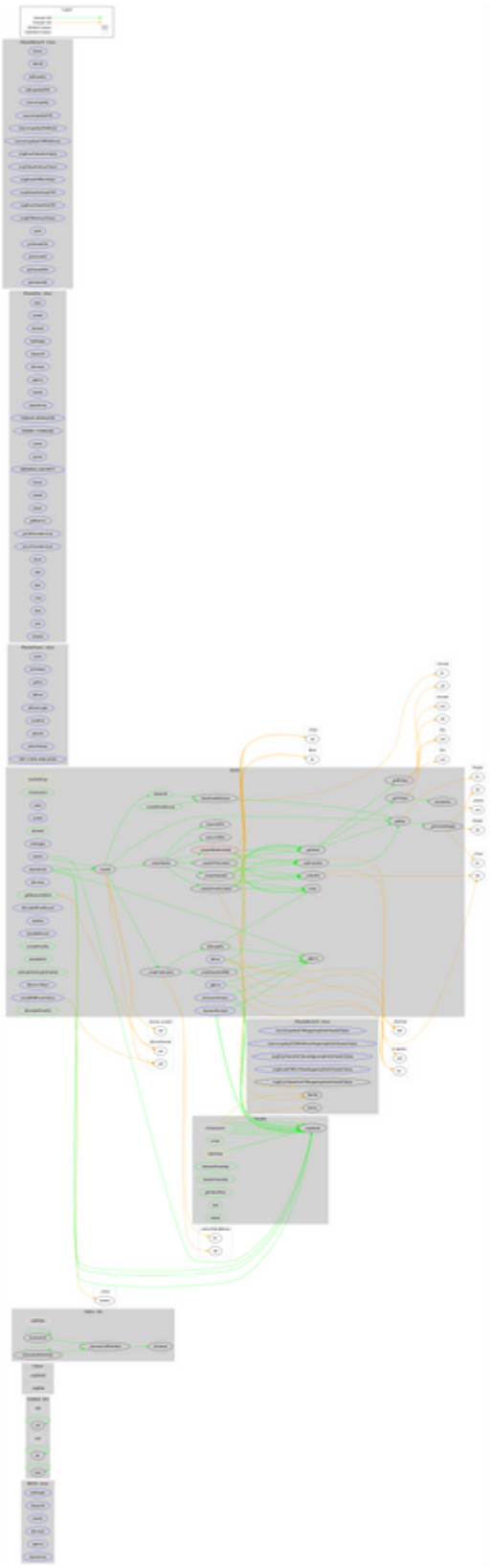
	_functionCallWithValue	Private	✓	
Ownable	Implementation	Context		
	<Constructor>	Internal	✓	
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	getUnlockTime	Public		-
	lock	Public	✓	onlyOwner
	unlock	Public	✓	-
IPancakeFactory	Interface			
	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-
	INIT_CODE_PAIR_HASH	External		-
IPancakePair	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	✓	-
IPancakeRouter01	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		-
IPancakeRouter02	Interface	IPancakeRouter01		
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
MANU	Implementation	Context, IBEP20, Ownable		
	<Constructor>	Public	✓	-
	name	External		-
	symbol	External		-
	decimals	Public		-
	totalSupply	External		-
	balanceOf	Public		-
	transfer	External	✓	-
	getBalanceAbleSell	Public		-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
	increaseAllowance	External	✓	-
	decreaseAllowance	External	✓	-
	isExcludedFromReward	External		-
	totalFees	External		-
	deliver	External	✓	-
	tokenFromReflection	Public		-
	excludeFromReward	Public	✓	onlyOwner
	includeInReward	External	✓	onlyOwner
	_transferBothExcluded	Private	✓	

	excludeFromFee	Public	✓	onlyOwner
	includeInFee	Public	✓	onlyOwner
	setSwapItAndLiquifyEnabled	Public	✓	onlyOwner
	<Receive Ether>	External	Payable	-
	rescueBNBFromContract	External	✓	onlyOwner
	_reflectFee	Private	✓	
	_getValues	Private		
	_getTValues	Private		
	_getRValues	Private		
	_getRate	Private		
	_getCurrentSupply	Private		
	_takeLiquidity	Private	✓	
	calculateFee	Private		
	removeAllFee	Private	✓	
	restoreAllFee	Private	✓	
	isExcludedFromFee	Public		-
	_approve	Private	✓	
	_transfer	Private	✓	
	_swapItAndLiquify	Private	✓	lockTheSwap
	swapTokensForBNB	Private	✓	
	addLiquidity	Private	✓	
	_tokenTransfer	Private	✓	
	_transferStandard	Private	✓	
	_transferToExcluded	Private	✓	
	_transferFromExcluded	Private	✓	

Contract Flow



Domain Info

Domain Name	marcoinu.io
Registry Domain ID	7a669fef3c4f4fcfb07d5f4e293973a3-DONUTS
Creation Date	2022-04-30T15:42:22Z
Updated Date	2022-04-30T16:00:02Z
Registry Expiry Date	2023-04-30T15:42:22Z
Registrar WHOIS Server	whois.namecheap.com
Registrar URL	https://www.namecheap.com/
Registrar	NameCheap, Inc.
Registrar IANA ID	1068

The domain has been created 5 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

The contract contains business logic inconsistencies that may break the token balance after some trades. The total supply is diverse from the address balances.

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