



Smart Contract Audit

FOR

MEME CEO

DATED : 29 Mar 23'



AUDIT SUMMARY

Project name – MEME CEO

Date: 29 March, 2023

Scope of Audit- Audit Ace was consulted to conduct the smart contract audit of the solidity source codes.

Audit Status: **Passed**

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	2	0	0	0
Acknowledged	0	0	0	0	0
Resolved	0	2	0	0	0

USED TOOLS

Tools:

1- Manual Review:

a line by line code review has been performed by audit ace team.

2- BSC Test Network:

all tests were done on BSC Test network, each test has its transaction has attached to it.

3- Slither : Static Analysis

Testnet Link: all tests were done using this contract, tests are done on BSC Testnet

<https://testnet.bscscan.com/address/0x11eae53f8f8f2f156dcbbc1c4b484b11dc68a6b8>



Token Information

Token Name : Meme CEO Token

Token Symbol: MCEO

Decimals: 9

Token Supply: 1,000,000,000

Token Address:

0x3E4eBa192A90a0665A0720245E2Dc96d58E11572

Checksum:

f21614208705e207da7e5cc2d712488775bd2916

Owner:

0xB4A898670309C16E3DB7828ab5c9BB983dbA648f



TOKEN OVERVIEW

Fees:

Buy Fees: 6%

Sell Fees: 6%

Transfer Fees: 6%

Fees Privilege: None

Ownership : Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Privileges: updating liquidity threshold -
excluding from fees - including in fees - including in
rewards - excluding from rewards



AUDIT METHODOLOGY

The auditing process will follow a routine as special considerations by Auditace:

- Review of the specifications, sources, and instructions provided to Auditace to make sure the contract logic meets the intentions of the client without exposing the user's funds to risk.
 - Manual review of the entire codebase by our experts, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
 - Specification comparison is the process of checking whether the code does what the specifications, sources, and instructions provided to Auditace describe.
 - Test coverage analysis determines whether the test cases are covering the code and how much code is exercised when we run the test cases.
 - Symbolic execution is analysing a program to determine what inputs cause each part of a program to execute.
 - Reviewing the codebase to improve maintainability, security, and control based on the established industry and academic practices.
-

VULNERABILITY CHECKLIST

- | | |
|--|---|
|  Return values of low-level calls |  Gasless Send |
|  Private modifier |  Using block.timestamp |
|  Multiple Sends |  Re-entrancy |
|  Using Suicide |  Tautology or contradiction |
|  Gas Limitand Loops |  Timestamp Dependence |
|  Address hardcoded |  Revert/require functions |
|  Exception Disorder |  Use of tx.origin |
|  Using inline assembly |  Integer overflow/underflow |
|  Divide before multiply |  Dangerous strict equalities |
|  Missing Zero Address Validation |  Using SHA3 |
|  Compiler version not fixed |  Using throw |
-



CLASSIFICATION OF RISK

Severity

Description

◆ Critical	These vulnerabilities could be exploited easily and can lead to asset loss, data loss, asset, or data manipulation. They should be fixed right away.
◆ High-Risk	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.
◆ Medium-Risk	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.
◆ Low-Risk	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.
◆ Gas Optimization /Suggestion	A vulnerability that has an informational character but is not affecting any of the code.

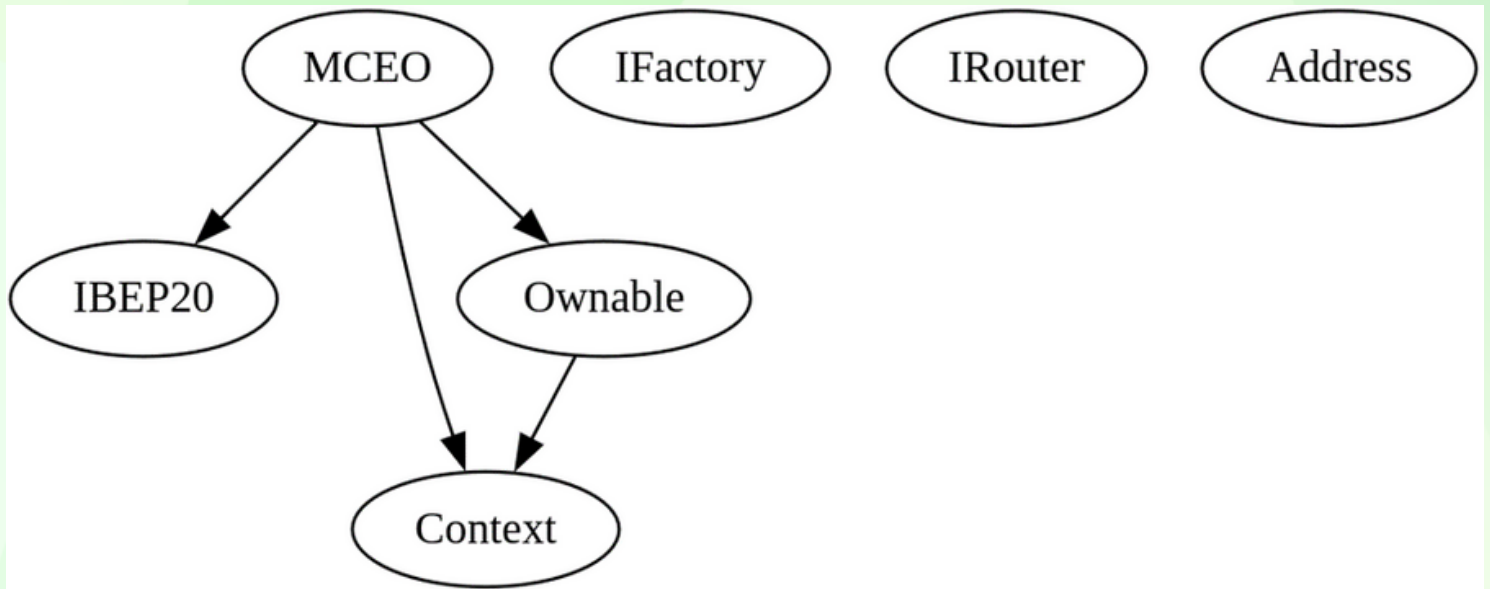
Findings

Severity

Found

◆ Critical	0
◆ High-Risk	2 (Resolved)
◆ Medium-Risk	0
◆ Low-Risk	0
◆ Gas Optimization / Suggestions	0

INHERITANCE TREE



POINTS TO NOTE

- **Owner is not able to modify buy/sell/transfer fees (6% each)**
 - **Owner must enable trading for investors to be able to trade**
 - **Owner is not able to set max buy/sell/transfer/hold amount**
 - **Owner is not able to blacklist an arbitrary wallet**
 - **Owner is not able to disable trades**
 - **Owner is not able to mint new tokens**
-

CONTRACT ASSESMENT






Contract	Type	Bases			
:-----: :-----: :-----: :-----: :-----:					
L	**Function Name**	**Visibility**	**Mutability**	**Modifiers**	
IBEP20 Interface					
L	totalSupply	External	!	NO!	
L	balanceOf	External	!	NO!	
L	transfer	External	!	NO!	
L	allowance	External	!	NO!	
L	approve	External	!	NO!	
L	transferFrom	External	!	NO!	
Context Implementation					
L	_msgSender	Internal	🔒		
L	_msgData	Internal	🔒		
Ownable Implementation Context					
L	<Constructor>	Public	!	NO!	
L	owner	Public	!	NO!	
L	renounceOwnership	Public	!	onlyOwner	
L	transferOwnership	Public	!	onlyOwner	
L	_setOwner	Private	🔒		
IFactory Interface					
L	createPair	External	!	NO!	
IRouter Interface					
L	factory	External	!	NO!	
L	WETH	External	!	NO!	
L	addLiquidityETH	External	!	NO!	
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External	!	NO!	
Address Library					
L	sendValue	Internal	🔒		
MCEO Implementation Context, IBEP20, Ownable					
L	<Constructor>	Public	!	NO!	
L	name	Public	!	NO!	
L	symbol	Public	!	NO!	
L	decimals	Public	!	NO!	
L	totalSupply	Public	!	NO!	
L	balanceOf	Public	!	NO!	

CONTRACT ASSESMENT

\angle	allowance	Public !		NO !
\angle	approve	Public !		NO !
\angle	transferFrom	Public !		NO !
\angle	increaseAllowance	Public !		NO !
\angle	decreaseAllowance	Public !		NO !
\angle	transfer	Public !		NO !
\angle	isExcludedFromReward	Public !		NO !
\angle	reflectionFromToken	Public !		NO !
\angle	EnableTrading	External !		onlyOwner
\angle	updatedDeadline	External !		onlyOwner
\angle	tokenFromReflection	Public !		NO !
\angle	excludeFromReward	Public !		onlyOwner
\angle	includeInReward	External !		onlyOwner
\angle	excludeFromFee	Public !		onlyOwner
\angle	includeInFee	Public !		onlyOwner
\angle	isExcludedFromFee	Public !		NO !
\angle	_reflectRfi	Private 		
\angle	_takeLiquidity	Private 		
\angle	_takeMarketing	Private 		
\angle	_takeOps	Private 		
\angle	_takeDev	Private 		
\angle	_getValues	Private 		
\angle	_getTValues	Private 		
\angle	_getRValues1	Private 		
\angle	_getRValues2	Private 		
\angle	_getRate	Private 		
\angle	_getCurrentSupply	Private 		
\angle	_approve	Private 		
\angle	_transfer	Private 		
\angle	_tokenTransfer	Private 		
\angle	swapAndLiquify	Private 		lockTheSwap
\angle	addLiquidity	Private 		
\angle	swapTokensForBNB	Private 		
\angle	bulkExcludeFee	External !		onlyOwner
\angle	updateMarketingWallet	External !		onlyOwner
\angle	updateDevWallet	External !		onlyOwner
\angle	updateOpsWallet	External !		onlyOwner
\angle	updateSwapTokensAtAmount	External !		onlyOwner
\angle	updateSwapEnabled	External !		onlyOwner
\angle	rescueBNB	External !		onlyOwner




CONTRACT ASSESMENT

| ^L | rescueAnyBEP20Tokens | Public  |  | onlyOwner |
| ^L | <Receive Ether> | External  |  | NO  |

| Symbol | Meaning |

|:-----:|-----|

|  | Function can modify state |

|  | Function is payable |

Token Distribution

it should be noted that the owner currently holds 100% of the total supply. However, information about the distribution of these tokens is not available, and it is recommended that investors exercise caution when considering this aspect.



STATIC ANALYSIS

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

MCE0.includeInReward(address) (contracts/Token.sol#417-428) has costly operations inside a loop:
- _excluded.pop() (contracts/Token.sol#424)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop

Context._msgData() (contracts/Token.sol#60-63) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

MCE0._rTotal (contracts/Token.sol#178) is set pre-construction with a non-constant function or state variable:
- (MAX - (MAX % _tTotal))
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#function-initializing-state

Pragma version^0.8.17 (contracts/Token.sol#21) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.19 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/Token.sol#140-150):
- (success) = recipient.call{value: amount}() (contracts/Token.sol#145)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function IRouter.WETH() (contracts/Token.sol#116) is not in mixedCase
Struct MCE0.valuesFromGetValues (contracts/Token.sol#215-229) is not in CapWords
Function MCE0.EnableTrading() (contracts/Token.sol#383-388) is not in mixedCase
Parameter MCE0.updatedDeadline(uint256)._deadline (contracts/Token.sol#390) is not in mixedCase
Parameter MCE0.updateSwapEnabled(bool)._enabled (contracts/Token.sol#807) is not in mixedCase
Parameter MCE0.rescueAnyBEP20Tokens(address,address,uint256)._tokenAddr (contracts/Token.sol#819) is not in mixedCase
Parameter MCE0.rescueAnyBEP20Tokens(address,address,uint256)._to (contracts/Token.sol#820) is not in mixedCase
Parameter MCE0.rescueAnyBEP20Tokens(address,address,uint256)._amount (contracts/Token.sol#821) is not in mixedCase
Constant MCE0._decimals (contracts/Token.sol#174) is not in UPPER_CASE_WITH_UNDERSCORES
Variable MCE0.genesis_block (contracts/Token.sol#182) is not in mixedCase
Constant MCE0._name (contracts/Token.sol#190) is not in UPPER_CASE_WITH_UNDERSCORES
Constant MCE0._symbol (contracts/Token.sol#191) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (contracts/Token.sol#61)" inContext (contracts/Token.sol#55-64)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

MCE0.lastSell (contracts/Token.sol#169) is never used in MCE0 (contracts/Token.sol#153-831)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

MCE0._tTotal (contracts/Token.sol#177) should be constant
MCE0.deadWallet (contracts/Token.sol#185) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

MCE0.pair (contracts/Token.sol#172) should be immutable
MCE0.router (contracts/Token.sol#171) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
```

Result => A static analysis of contract's source code has been performed using slither,

No major issues were found in the output



FUNCTIONAL TESTING

Router (PCS V2):

0xD99D1c33F9fC3444f8101754aBC46c52416550D1

All the functionalities have been tested, no issues were found

1- Adding liquidity (passed):

<https://testnet.bscscan.com/tx/0x255cb2f03ca9d6b82c4795089474377784e24fe887562d69b272a0bc14f8130f>

2- Buying when excluded (0% tax) (passed):

<https://testnet.bscscan.com/tx/0x3ca510f550e430d71e9db213230b828c61745132366f3a836817e5618a4c2d8a>

3- Selling when excluded (0% tax) (passed):

<https://testnet.bscscan.com/tx/0x4b0a732257c7c1af3ce76f0e35cc92be465436bf0169ff5c9500d522b7835c6b>

4- Transferring when excluded (0% tax) (passed):

<https://testnet.bscscan.com/tx/0xd9ccafda56307e7a5eeb0e095477279680d80797bde110cfa6a882ae19413200>

5- Buying when not excluded (6% tax) (passed):

<https://testnet.bscscan.com/tx/0x0466adf5629402886318f1793c0eb5c3636b0506337274201bccf66ca096b233>

6- Selling when not excluded (6% tax) (passed):

<https://c.bscscan.com/tx/0xfcd968a6ba3684bfe83e88fe403834e40dfe29f0ef6334b1098b194c436d13e6>

FUNCTIONAL TESTING

7- Transferring when not excluded (1% tax) (passed):

<https://testnet.bscscan.com/tx/0x65b7c0c18617009f0cd79960d74ea2128d068312a060299952804e0aea868417>

8- Internal swap (passed):

Marketing wallet received ETH

<https://testnet.bscscan.com/tx/0xfcd968a6ba3684bfe83e88fe403834e40dfe29f0ef6334b1098b194c436d13e6>

9- Auto Liquidity (passed):

```
https://testnet.bscscan.com/token/0xf9013229cee46bb22d26274  
420a08c848596324b?  
a=0x00000000000000000000000000000000000000000000dead
```

MANUAL TESTING

Centralization - Owner must enable trading

Severity: **High**

Function: EnableTrading

Lines: 328

Status: **Resolved**

Overview:

The owner must activate trading for investors to buy, sell, or transfer tokens. If trading remains disabled, token holders will be unable to trade their tokens.

```
function EnableTrading() external onlyOwner {  
    require(!tradingEnabled, "Cannot re-enable trading");  
    tradingEnabled = true;  
    swapEnabled = true;  
    genesis_block = block.number;  
}
```

Recommendation:

Incorporate a safety mechanism that allows investors to activate trading if a specified duration has elapsed since the conclusion of the presale.

Since contract is owned by safu dev, enabling trades is guaranteed.

MANUAL TESTING

Logical – Setting swap threshold to 0

Severity: High

Function: updateSwapTokensAtAmount

Lines: 761

Status: not resolved

Overview:

setting swap threshold to 0 can disable sells if contract balance is more than threshold.

```
function updateSwapTokensAtAmount(uint256 amount) external onlyOwner {
    require(
        amount <= 1e7,
        "Cannot set swap threshold amount higher than 1% of tokens"
    );
    swapTokensAtAmount = amount * 10 ** decimals;
}
```

Recommendation:

ensure that swap threshold can not be zero.

Since contract is owned by safu dev, swap threshold will not be set to 0

Social Media Overview

**Here are the Social Media Accounts of
Meme CEO**



<https://t.me/MemeCEOX100>



<https://twitter.com/MemeCEOX100>



<https://memeceo.cc/>



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. 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 Auditace 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 Auditace 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 Auditace team disclaims any liability for the resulting losses.



ABOUT AUDITACE

We specializes in providing thorough and reliable audits for Web3 projects. With a team of experienced professionals, we use cutting-edge technology and rigorous methodologies to evaluate the security and integrity of blockchain systems. We are committed to helping our clients ensure the safety and transparency of their digital assets and transactions.



<https://auditace.tech/>



https://t.me/Audit_Ace



https://twitter.com/auditace_



<https://github.com/Audit-Ace>
