AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

CHIHUAHUACEO

DATED : 05 Mar 23'

AUDIT SUMMARY

Project name – CHIHUAHUACEO

Date: 05 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	0	0	1	0
Acknowledged	0	0	0	0	0
Resolved	0	0	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/0x3745cF8a2e38B268710CC7C04689491E885c7350#code

Token Information

Token Name : Chihuahua CEO

Token Symbol: Chihuahua CEO

Decimals: 9

Token Supply: 420,000,000,000,000,000

Token Address:
0x8d2CfC20c20de53934cD1657a9187f46591dDa31

Checksum:
d191eb77831c39a6828fd406419d233bdf221735

Owner:
0xBCf84E37F83a55a03a3BE9f7fa2c52f95ecA5509

TOKEN OVERVIEW

Fees:

Buy Fees: 10%

Sell Fees: 10%

Transfer Fees: 10%

Fees Privilige: None

Ownership : Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Priviliges: including and excluding from fees

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

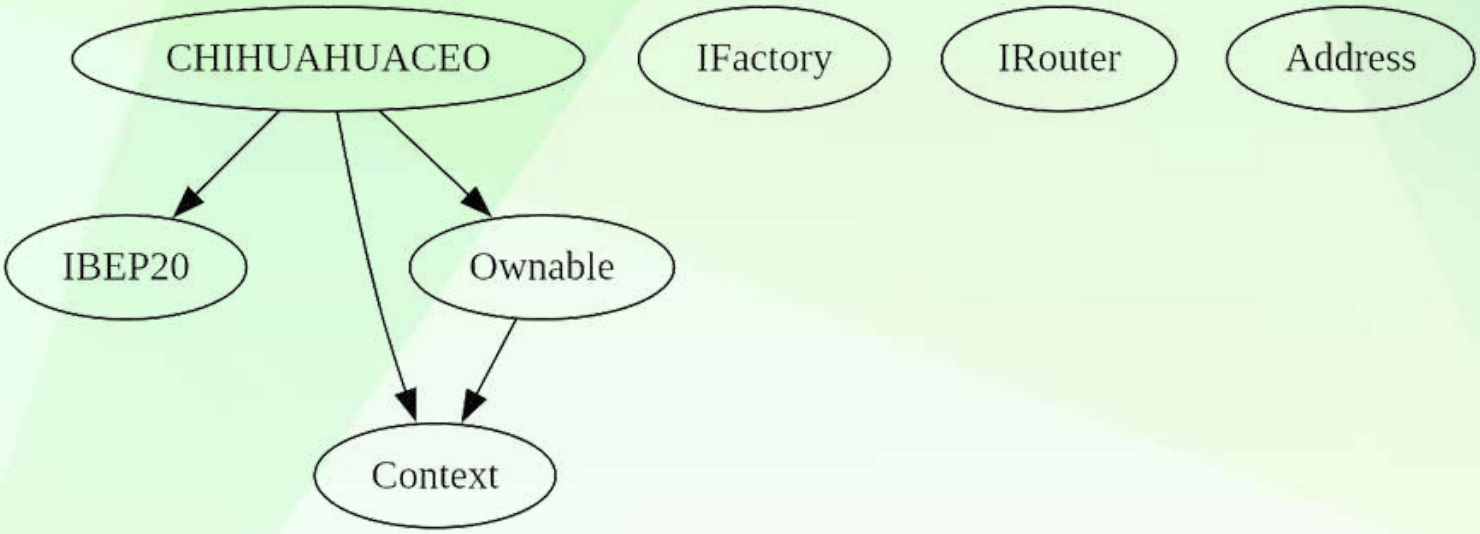
CLASSIFICATION OF RISK

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	0
Medium-Risk	0
Low-Risk	1
Gas Optimization / Suggestions	0

INHERITANCE TREE



POINTS TO NOTE

- Owner is not able to change fees (10% for buy/sell/transfers static)
- 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

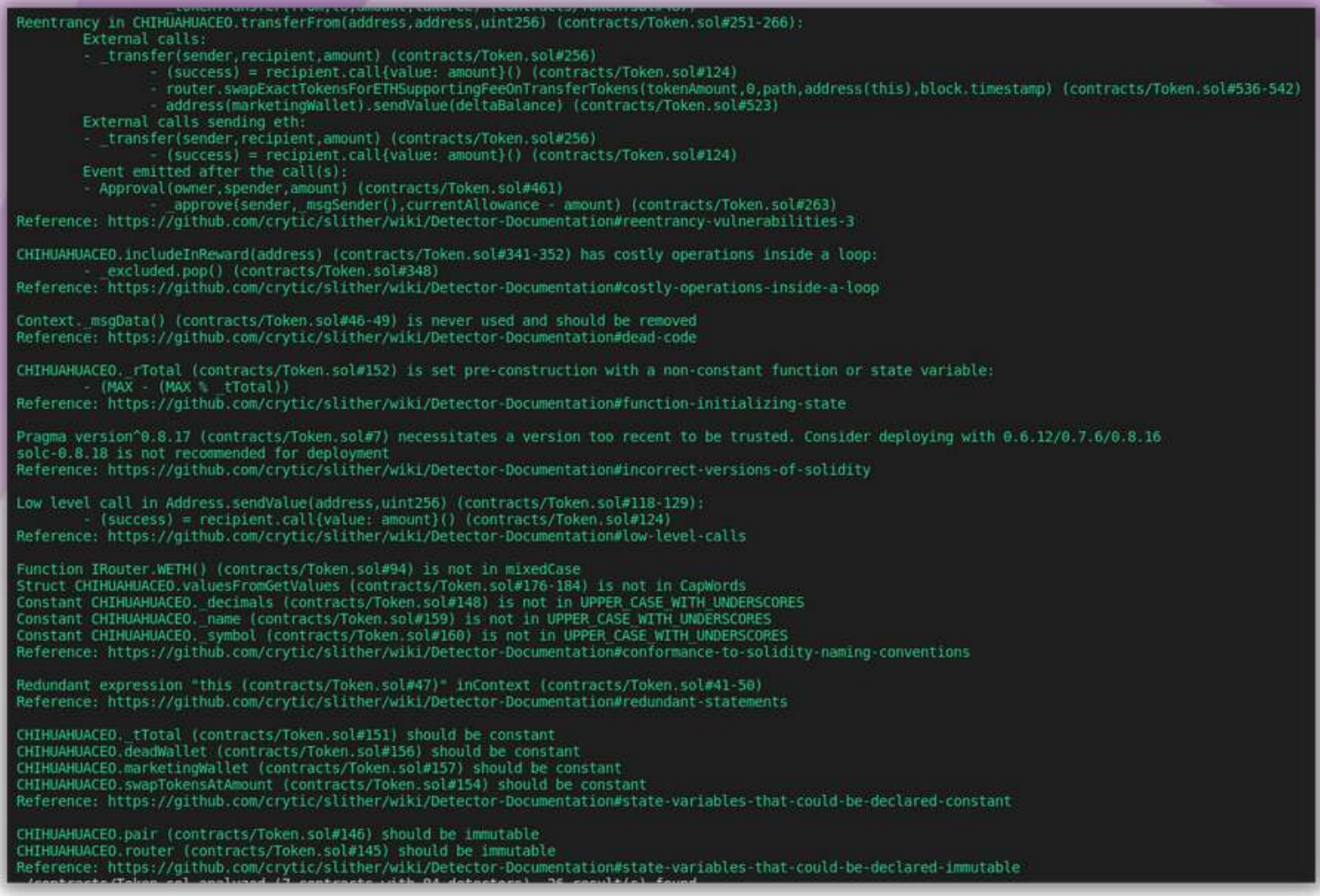
Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
IBEP20	Interface				
└	totalSupply	External ❗		NO ❗
└	balanceOf	External ❗		NO ❗
└	transfer	External ❗	🛑	NO ❗
└	allowance	External ❗		NO ❗
└	approve	External ❗	🛑	NO ❗
└	transferFrom	External ❗	🛑	NO ❗
Context	Implementation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

CONTRACT ASSESMENT

└	transferFrom	Public ❗	🛑	NO ❗
└	increaseAllowance	Public ❗	🛑	NO ❗
└	decreaseAllowance	Public ❗	🛑	NO ❗
└	transfer	Public ❗	🛑	NO ❗
└	isExcludedFromReward	Public ❗		NO ❗
└	reflectionFromToken	Public ❗		NO ❗
└	tokenFromReflection	Public ❗		NO ❗
└	excludeFromReward	Public ❗	🛑	onlyOwner
└	includeInReward	External ❗	🛑	onlyOwner
└	excludeFromFee	Public ❗	🛑	onlyOwner
└	includeInFee	Public ❗	🛑	onlyOwner
└	isExcludedFromFee	Public ❗		NO ❗
└	_reflectRfi	Private 🔐	🛑	
└	_takeMarketing	Private 🔐	🛑	
└	_getValues	Private 🔐		
└	_getTValues	Private 🔐		
└	_getRValues	Private 🔐		
└	_getRate	Private 🔐		
└	_getCurrentSupply	Private 🔐		
└	_approve	Private 🔐	🛑	
└	_transfer	Private 🔐	🛑	
└	_tokenTransfer	Private 🔐	🛑	
└	swapAndLiquify	Private 🔐	🛑	lockTheSwap
└	swapTokensForBNB	Private 🔐	🛑	
└	bulkExcludeFee	External ❗	🛑	onlyOwner
└	<Receive Ether>	External ❗	💵	NO ❗

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS



```
Reentrancy in CHIHUAHUACEO.transferFrom(address,address,uint256) (contracts/Token.sol#251-266):
        External calls:
        - _transfer(sender,recipient,amount) (contracts/Token.sol#256)
                - (success) = recipient.call{value: amount}() (contracts/Token.sol#124)
                - router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (contracts/Token.sol#536-542)
                - address(marketingWallet).sendValue(deltaBalance) (contracts/Token.sol#523)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (contracts/Token.sol#256)
                - (success) = recipient.call{value: amount}() (contracts/Token.sol#124)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (contracts/Token.sol#461)
                - _approve(sender,_msgSender(),currentAllowance - amount) (contracts/Token.sol#263)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

CHIHUAHUACEO.includeInReward(address) (contracts/Token.sol#341-352) has costly operations inside a loop:
        - _excluded.pop() (contracts/Token.sol#348)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop

Context._msgData() (contracts/Token.sol#46-49) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

CHIHUAHUACEO._rTotal (contracts/Token.sol#152) 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#7) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.18 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#118-129):
        - (success) = recipient.call{value: amount}() (contracts/Token.sol#124)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function IRouter.WETH() (contracts/Token.sol#94) is not in mixedCase
Struct CHIHUAHUACEO.valuesFromGetValues (contracts/Token.sol#176-184) is not in CapWords
Constant CHIHUAHUACEO._decimals (contracts/Token.sol#148) is not in UPPER_CASE_WITH_UNDERSCORES
Constant CHIHUAHUACEO._name (contracts/Token.sol#159) is not in UPPER_CASE_WITH_UNDERSCORES
Constant CHIHUAHUACEO._symbol (contracts/Token.sol#160) 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#47)" inContext (contracts/Token.sol#41-50)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

CHIHUAHUACEO._tTotal (contracts/Token.sol#151) should be constant
CHIHUAHUACEO.deadWallet (contracts/Token.sol#156) should be constant
CHIHUAHUACEO.marketingWallet (contracts/Token.sol#157) should be constant
CHIHUAHUACEO.swapTokensAtAmount (contracts/Token.sol#154) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

CHIHUAHUACEO.pair (contracts/Token.sol#146) should be immutable
CHIHUAHUACEO.router (contracts/Token.sol#145) 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/0xa19868813f5c545869658164b832c6e04c8d941275506193c9c534e80d8921c5

2- Buying when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x0691cc2279868f9b794e6a7c11f649a936773175d71e2bd55fc0cf1f135c8e10

3- Selling when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0xed51a537ce40af475369ef5eb5895124c3b546becd1ea5627046ca52b1430415

4- Transferring when transfer without fee enabled (0% tax) (passed):
https://testnet.bscscan.com/tx/0x1ea53faa1eea5b271480f9a64159ea9a3fc70c86aaa69b42130a5ca637da778b

5- Buying when not excluded (10% tax) (passed):
https://testnet.bscscan.com/tx/0xd4368627af8c5e5f378b986bdb0b66ec29d22444d862dc59a27e0f2c085dc6cf

6- Selling when not excluded (10% tax) (passed):
https://testnet.bscscan.com/tx/0xdca4f61da2e3132d3c35e75bdcb86cad128b8fa243fbaf322ea782b76ee725b1

FUNCTIONAL TESTING

7- Transferring when transfer without fee disabled (10% tax) (passed):
https://testnet.bscscan.com/tx/0x09f1a6b835b6bfa106b8e3c869ac81caeec2667f42816ae928b6434e01487aa1

8- Internal swap (passed):
marketing wallet received ETH
https://testnet.bscscan.com/address/0xeac74d8b58cae17193b2dfb86ae3f5efd42416c0#internaltx

MANUAL TESTING

Issue: no transferOwnership Function
Type : Logical
Function: ---
Line: 42-69
Severity: Low

Overview:
the contract does not have a transferOwnership function, which means that the ownership of the contract cannot be transferred to another address. This can be a problem if the original owner loses control of their private keys or if they are no longer able to manage the contract for any reason

Recommendations
To address the issue identified in this audit, we recommend the following:
1.Implement a TransferOwnership Function The contract should be updated to include a transferOwnership function that allows the current owner to transfer ownership of the contract to another address. This function should include proper access control to prevent unauthorized transfers of ownership.

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