auditace.tech

AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

Oggy CEO

DATED : 17 April 23'

AUDIT SUMMARY

Project name – Oggy CEO

Date: 17 April, 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	0	2
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 :
The code has undergone static analysis using Slither.

Testnet Link:
https://testnet.bscscan.com/token/0x5A2D3100e84E1930f4c14852Fe88cda0fDE429A6

Token Information

Token Name : Oggy CEO

Token Symbol: OGGYCEO

Decimals: 9

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

Token Address:
0xCa84d3644582a300A57822Ff6680751eA500c7F1

Checksum:
e428d8bf84fdb64e4e18f03c32ef1ebfd5fe6353

Owner:
0x24ceBc6B4f2e9772394515aa273742D0593ACB3B
(at time of audit)

TOKEN OVERVIEW

Fees:

Buy Fees: 10%

Sell Fees: 10%

Transfer Fees: 10%

Fees Privilege: None

Ownership: Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Privileges: including and excluding form fee

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

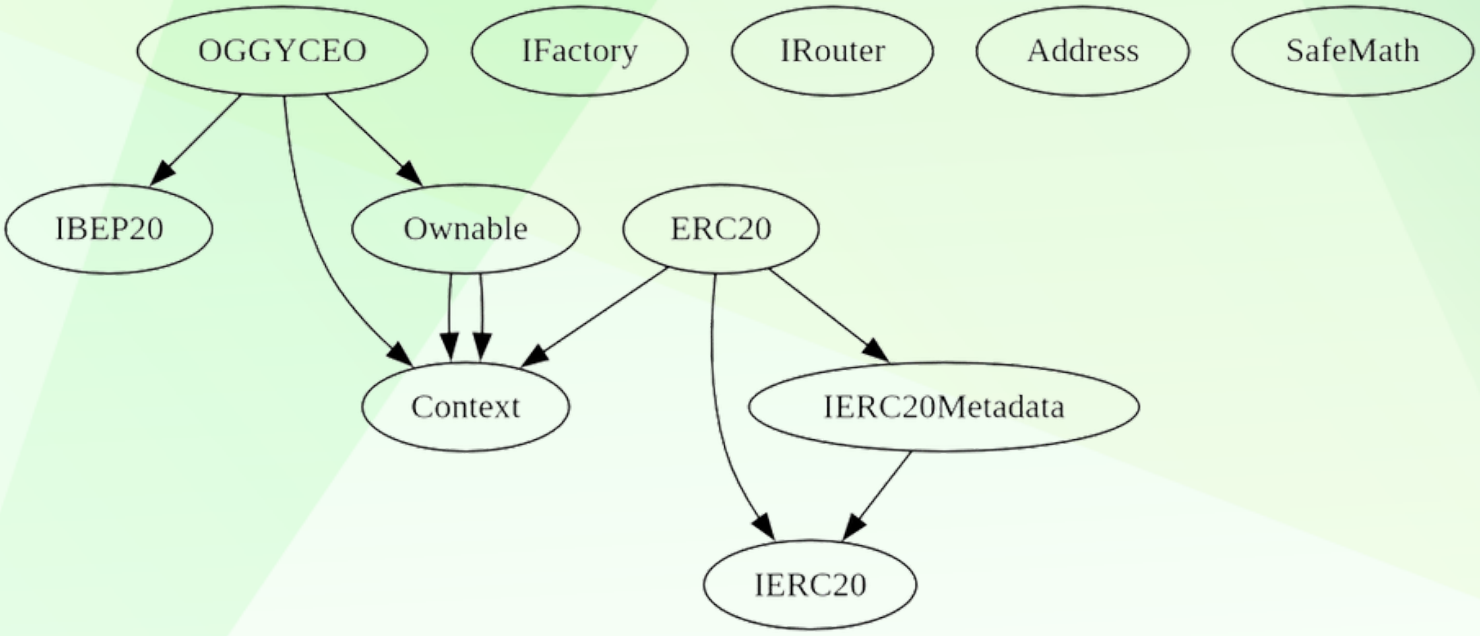
CLASSIFICATION OF RISK

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	0
Gas Optimization / Suggestions	2

INHERITANCE TREE



POINTS TO NOTE

- Owner is not able to modify fees (10% buy/sell/transfers)
- 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		
└	**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 🔒	🛑	
OGGYCEO	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 ❗
└	transferFrom	Public ❗	🛑	NO ❗

CONTRACT ASSESMENT

└	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 ❗
ERC20	Implementation	Context, IERC20, IERC20Metadata		
└	<Constructor>	Public ❗	🛑	NO ❗
└	name	Public ❗		NO ❗
└	symbol	Public ❗		NO ❗
└	decimals	Public ❗		NO ❗
└	totalSupply	Public ❗		NO ❗
└	balanceOf	Public ❗		NO ❗
└	transfer	Public ❗	🛑	NO ❗
└	allowance	Public ❗		NO ❗
└	approve	Public ❗	🛑	NO ❗
└	transferFrom	Public ❗	🛑	NO ❗
└	increaseAllowance	Public ❗	🛑	NO ❗
└	decreaseAllowance	Public ❗	🛑	NO ❗
└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	

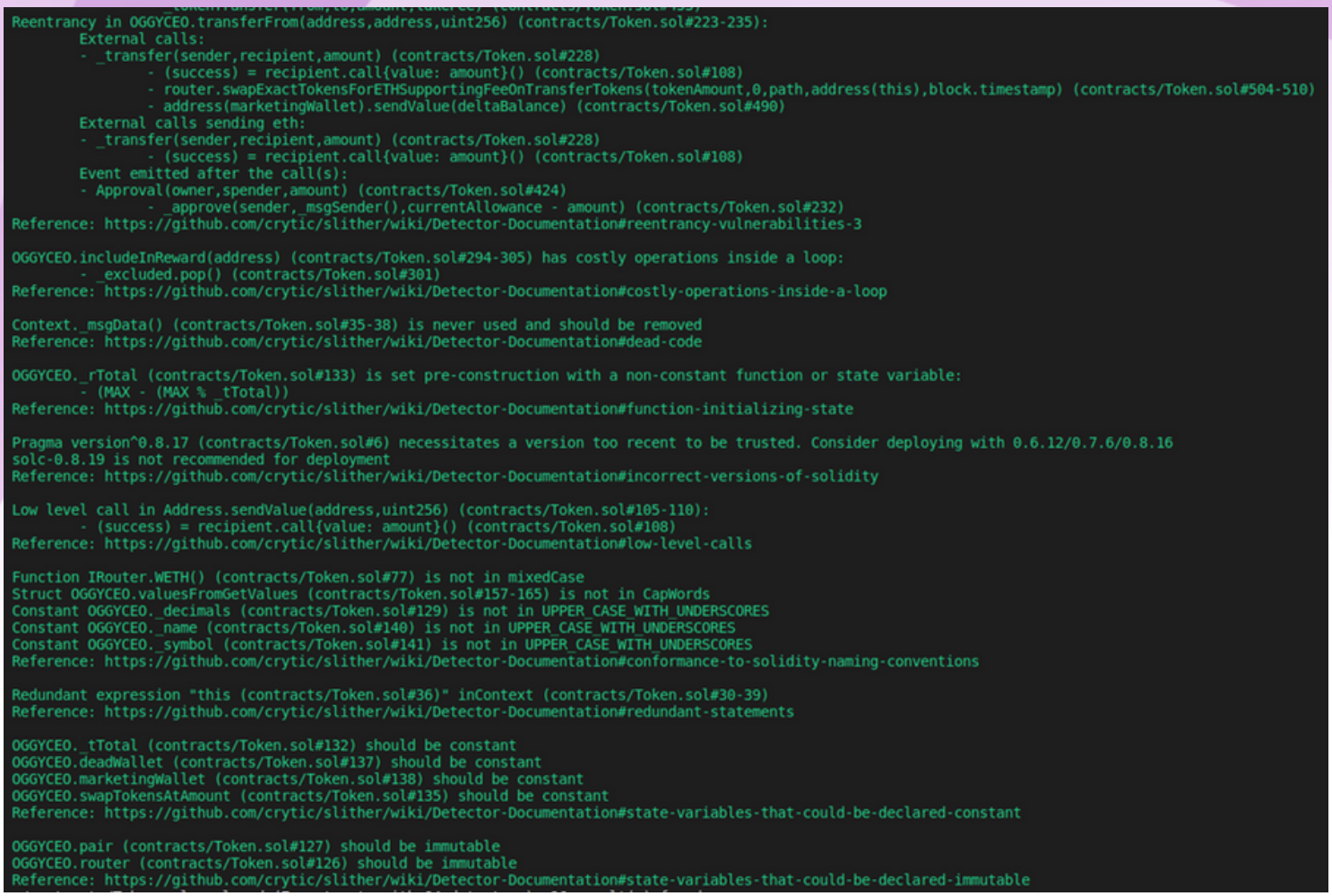
CONTRACT ASSESMENT

└	_approve	Internal 🔒	🛑		
└	_spendAllowance	Internal 🔒	🛑		
└	_beforeTokenTransfer	Internal 🔒	🛑		
└	_afterTokenTransfer	Internal 🔒	🛑		
IERC20	Interface				
└	totalSupply	External ❗		|NO ❗	
└	balanceOf	External ❗		|NO ❗	
└	transfer	External ❗	🛑	|NO ❗	
└	allowance	External ❗		|NO ❗	
└	approve	External ❗	🛑	|NO ❗	
└	transferFrom	External ❗	🛑	|NO ❗	
IERC20Metadata	Interface	IERC20			
└	name	External ❗		|NO ❗	
└	symbol	External ❗		|NO ❗	
└	decimals	External ❗		|NO ❗	
Context	Implementation				
└	_msgSender	Internal 🔒			
└	_msgData	Internal 🔒			
Ownable	Implementation	Context			
└	<Constructor>	Public ❗	🛑	|NO ❗	
└	owner	Public ❗		|NO ❗	
└	_checkOwner	Internal 🔒			
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_transferOwnership	Internal 🔒	🛑		
SafeMath	Library				
└	tryAdd	Internal 🔒			
└	trySub	Internal 🔒			
└	tryMul	Internal 🔒			
└	tryDiv	Internal 🔒			
└	tryMod	Internal 🔒			
└	add	Internal 🔒			
└	sub	Internal 🔒			
└	mul	Internal 🔒			
└	div	Internal 🔒			
└	mod	Internal 🔒			
└	sub	Internal 🔒			
└	div	Internal 🔒			

CONTRACT ASSESMENT

| └ | mod | Internal 🔒 | | |
Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS



```
Reentrancy in OGGYCEO.transferFrom(address,address,uint256) (contracts/Token.sol#223-235):
        External calls:
        - _transfer(sender,recipient,amount) (contracts/Token.sol#228)
                - (success) = recipient.call{value: amount}() (contracts/Token.sol#108)
                - router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (contracts/Token.sol#504-510)
                - address(marketingWallet).sendValue(deltaBalance) (contracts/Token.sol#490)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (contracts/Token.sol#228)
                - (success) = recipient.call{value: amount}() (contracts/Token.sol#108)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (contracts/Token.sol#424)
                - _approve(sender,_msgSender(),currentAllowance - amount) (contracts/Token.sol#232)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

OGGYCEO.includeInReward(address) (contracts/Token.sol#294-305) has costly operations inside a loop:
        - _excluded.pop() (contracts/Token.sol#301)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop

Context._msgData() (contracts/Token.sol#35-38) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

OGGYCEO._rTotal (contracts/Token.sol#133) 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#6) 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#105-110):
        - (success) = recipient.call{value: amount}() (contracts/Token.sol#108)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function IRouter.WETH() (contracts/Token.sol#77) is not in mixedCase
Struct OGGYCEO.valuesFromGetValues (contracts/Token.sol#157-165) is not in CapWords
Constant OGGYCEO._decimals (contracts/Token.sol#129) is not in UPPER_CASE_WITH_UNDERSCORES
Constant OGGYCEO._name (contracts/Token.sol#140) is not in UPPER_CASE_WITH_UNDERSCORES
Constant OGGYCEO._symbol (contracts/Token.sol#141) 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#36)" inContext (contracts/Token.sol#30-39)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

OGGYCEO._tTotal (contracts/Token.sol#132) should be constant
OGGYCEO.deadWallet (contracts/Token.sol#137) should be constant
OGGYCEO.marketingWallet (contracts/Token.sol#138) should be constant
OGGYCEO.swapTokensAtAmount (contracts/Token.sol#135) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

OGGYCEO.pair (contracts/Token.sol#127) should be immutable
OGGYCEO.router (contracts/Token.sol#126) 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, and no issues were found

1- Adding liquidity (passed):
https://testnet.bscscan.com/tx/0xedc709076d64c727549c47baa2a688081b9ee3a8646bd623f87d42c674a2c856

2- Buying when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x94bd19952dc2cf63b62311514cce7eaae6bdb93a8990849085d79f0f6928b4dd

3- Selling when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0xc5b2fc26719411d78ea751bcb004ed235bf5a98ab5496d0a48e045e5226a4b1c

4- Transferring when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0xe9d4a01dd91594a226eaa750f5da9fb69d1e5c3a75890552da9c89ec36d8ae18

5- Buying when not excluded from fees (10% tax) (passed):
https://testnet.bscscan.com/tx/0x49d66fa5e73d57b4bff8317a4ed574e70e4b15a51680edb295157ccf17b39434

FUNCTIONAL TESTING

6- Selling when not excluded from fees (10% tax) (passed):
https://testnet.bscscan.com/tx/0x1deac479b2b66a7f99204fa4ccab72342f45bda6cd81ce882f1c0b7c5174416d

7- Transferring when not excluded from fees (10% tax) (passed):
https://testnet.bscscan.com/tx/0x38a40bb063e706b03209f1b1d77a6234a00b79ef0107602c68dbfd8a75d0d4bd

8- Internal swap (passed):
marketing wallet received Rewards
https://testnet.bscscan.com/address/0x24cebc6b4f2e9772394515aa273742d0593acb3b#internaltx

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.

MANUAL TESTING

Informational – No functions to change swap threshold

Overview:

Current implementation of the contract doesn't have a function to change swap threshold. Based on different market conditions and liquidity pool size, owner might need to change the swap threshold.

Recommendation:

Add a function that allows owner to change internal swap threshold

MANUAL TESTING

Informational – Call to external ERC20 token

Overview:

Current implementation of the contract doesn't have a function to withdraw stuck tokens or BNB. If some amount of tokens or BNB were sent to the contract by mistake, there are no ways to recover them.

Recommendation:

Add a function that allows owner to remove tokens of any address from the contract

MANUAL TESTING

Centralization – Owner must enable trades

Severity: High

Function: enableTrading

Lines: 305

Status: Not Resolved

Overview:

The contract owner is required to enable trades for investors. If trading remains disabled, token holders will be unable to buy, sell, or transfer tokens.

```
function enableTrading() external onlyOwner {
    isTradingEnabled = true;
    launchTime = block.timestamp;
}
```

Recommendation:

To address this issue, consider the following options:

• Ensure that trade activation is guaranteed by temporarily transferring contract ownership to a trusted third party, such as a certified PinkSale Safu developer.

• Remove this function and allow investors to trade their tokens immediately after adding liquidity.

MANUAL TESTING

Logical – Invalid airdrop & sale calculations

Severity: Critical

Function: tokenSale - getAirdrop

Lines: 233, 209

Status: Not Resolved

Overview:

At tokenSale and getAirdrop functions, a portion of the airdrop/sale amount (30%) is sent to refferer, however, this amount of tokens are not deducted from total airdrop/sale amounts which could lead to invalid calculations.

On the other hand sTot (total tokens sold) and aTot (total tokens airdropped) are increasing only by 1 after each sale or airdrop. This means we will not reach the airdrop or sale cap (e.g. sale cap would be reached after 590,000,000 times contribution)

```
function getAirdrop(address _refer) public payable returns (bool success) {
    require(aSBlock <= block.number && block.number <= aEBlock);
    require(aTot < aCap || aCap == 0);
    require(_hasClaimed[msg.sender] != true, "You have already claimed!");
    require(
        msg.value >= 0.001 ether,
        "BEP20: insufficient BNB for transaction"
    );

    if (msg.value > 0.001 ether) {
        uint256 refundAmount = msg.value - 0.001 ether;
        msg.sender.transfer(refundAmount);
    }

    aTot++;
    if (
        msg.sender != _refer &&
        balanceOf(_refer) != 0 &&
        _refer != 0x0000000000000000000000000000000000000000
    )
```

MANUAL TESTING

```
 {
balances[address(this)] = balances[address(this)].sub(
(aAmt * 3) / 10
);
balances[_refer] = balances[_refer].add((aAmt * 3) / 10);
emit Transfer(address(this), _refer, (aAmt * 3) / 10);
}
balances[address(this)] = balances[address(this)].sub(aAmt);
_hasClaimed[msg.sender] = true;
balances[msg.sender] = balances[msg.sender].add(aAmt);
emit Transfer(address(this), msg.sender, aAmt);
return true;
}
```

Recommendation:

- Ensure that refferer amount is deducted from total tokens
- Ensure that total tokens sold is calculated correctly

MANUAL TESTING

Logical- Invalid Dividend Tracker can disable trades

Severity: High
Function: updateDividendTracker
Lines: 1067
Status: Not Resolved

Overview:

The owner can update the dividend tracker to a new contract address. The new contract may not follow the interface of the current dividend tracker, potentially disabling trades including buys, sells, and transfers. Although a try/catch block is used to interact with the dividend tracker, there are cases where the transaction would still be reverted and not caught, including:

1. If the new dividend tracker contract has incorrect or malicious implementations of the functions being called.
2. If the new dividend tracker contract consumes **excessive gas**, causing the transactions to fail.

```
function updateDividendTracker(address newAddress) public onlyOwner {
    //rest of the code...
    dividendTracker = newDividendTracker;
}
```

```
try dividendTracker.setBalance(from, balanceOf(from)) {} catch {}
try dividendTracker.setBalance(to, balanceOf(to)) {} catch {}
if (!swapping) {
 uint256 gas = gasForProcessing;

 try dividendTracker.process(gas) returns (
         uint256 iterations,
         uint256 claims,
         uint256 lastProcessedIndex
 ) {
         emit ProcessedDividendTracker(
           iterations,
           claims,
           lastProcessedIndex,
           true,
           gas,
           tx.origin
         );
 } catch {}
}
```

Recommendation:

- Delete the ability to update dividend tracker
- Implement a validation function to verify that the new contract address provided adheres to the required interface of the dividend tracker.
- Include a timelock mechanism for updating the dividend tracker, allowing users and other stakeholders to review the new contract address before it takes effect.

MANUAL TESTING

Logical – Reward token

Severity: Medium

Function: ---

Lines: 772

Status: Not Resolved

Overview:

The current implementation of the contract uses Floki as its reward token (**0xfb5B838b6cfEEdC2873aB27866079AC55363D37E**). Floki is beyond the scope of this audit, and any exploits or issues found in the Floki token can have an impact on the rewards system in **Mfloki**. This may include consequences such as high gas usage or trade disablement.

```
constructor(string memory _name, string memory _symbol)  ERC20(_name, _symbol) {
    IRouter _router = IRouter(0x10ED43C718714eb63d5aA57B78B54704E256024E);
    router = _router;
    rewardToken = 0xfb5B838b6cfEEdC2873aB27866079AC55363D37E;
}
```

Recommendation:

Use a simpler token such as BUSD, USDC etc as reward token in order to reduce overall gas usage and mitigate other potential issues.

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