auditace.tech

AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

Mrs Floki

DATED : 16 Apr 23'

AUDIT SUMMARY

Project name – Mrs Floki

Date: 16 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	2	1	0	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/token/0x8c894d73d90d383aed1cac1e54e5559c7cb86e23#code

Token Information

Token Name : Mrs Floki

Token Symbol: MFloki

Decimals: 9

Token Supply: 1,000,000,000,000,000

Token Address:
0x6f7BbEc92324D359DCa4754c0E7026b401C4C369

Checksum:
07e4cb58d75dab187fbbcb0134b8b8c89da70ab8

Owner:
0x74133652573C748dBc9ED66BAf5F944783314255
(at time of audit)

Deployer:
0x9ff5037b13Df356A3737097c8f16712F4EA864D0

TOKEN OVERVIEW

Fees:

Buy Fees: up to 12%

Sell Fees: up to 12%

Transfer Fees: 0%

Fees Privilige: Owner

Ownership : Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Priviliges: including and excluding form fee - changing swap threshold - modifying 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

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

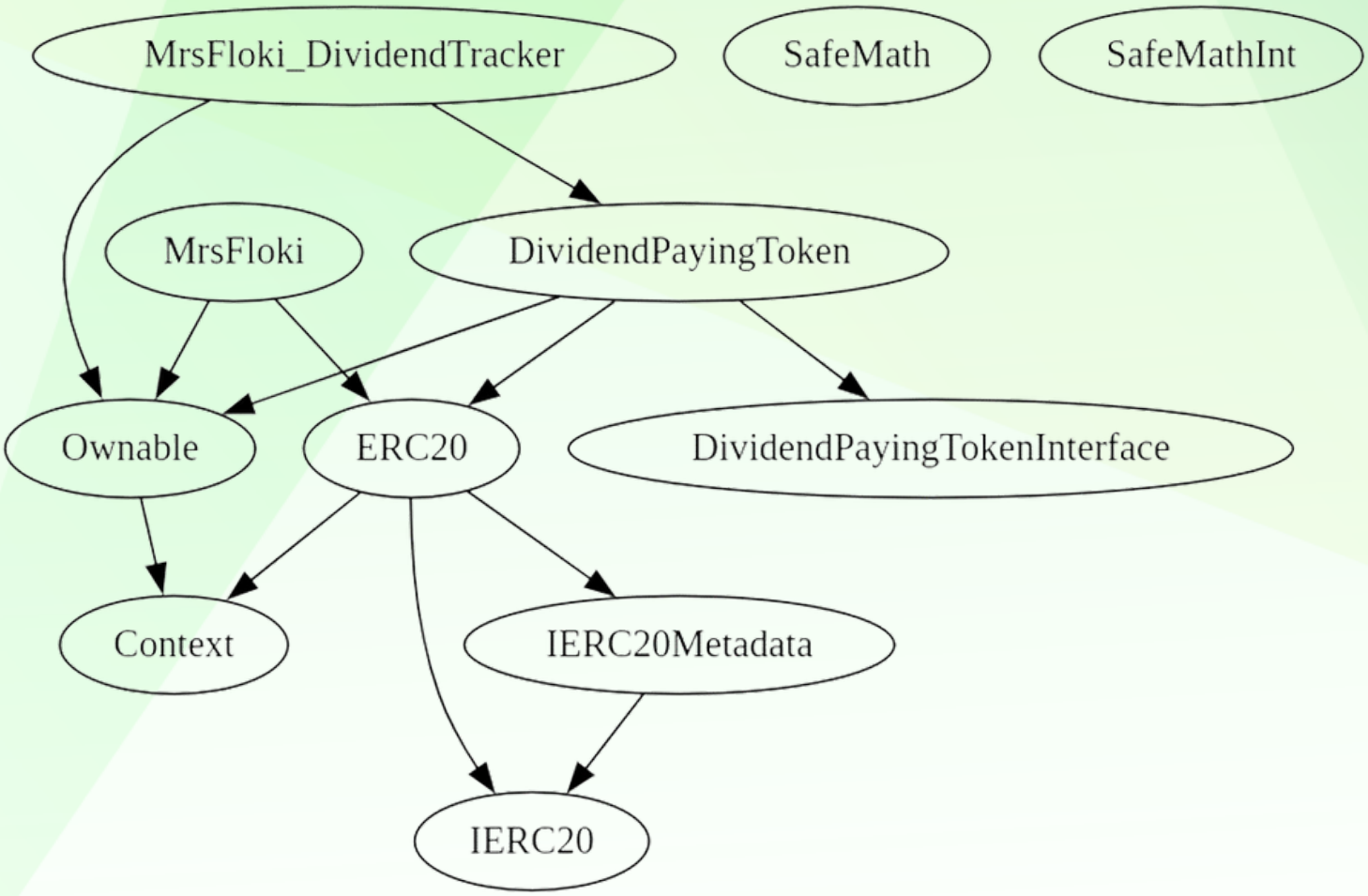
CLASSIFICATION OF RISK

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

INHERITANCE TREE



POINTS TO NOTE

- Owner is able to set buy/sell fees more than 12% each
- Owner is not ablet o set transfer taxes (0% forever)
- 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**
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	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❗
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 🔒	🛑	
└	_tokengeneration	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
SafeMath	Library			
└	add	Internal 🔒		

CONTRACT ASSESMENT

└	sub	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	mod	Internal 🔒		
SafeMathInt	Library			
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	sub	Internal 🔒		
└	add	Internal 🔒		
└	abs	Internal 🔒		
└	toUint256Safe	Internal 🔒		
SafeMathUint	Library			
└	toInt256Safe	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO❗
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
IPair	Interface			
└	sync	External ❗	🛑	NO❗
IFactory	Interface			
└	createPair	External ❗	🛑	NO❗
└	getPair	External ❗		NO❗
IRouter	Interface			
└	factory	External ❗		NO❗
└	WETH	External ❗		NO❗
└	addLiquidityETH	External ❗	💵	NO❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
└	swapExactETHForTokens	External ❗	💵	NO❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
DividendPayingTokenInterface	Interface			
└	dividendOf	External ❗		NO❗

CONTRACT ASSESMENT

└	distributeDividends	External ❗	💵	NO❗
└	withdrawableDividendOf	External ❗		|NO❗
└	withdrawnDividendOf	External ❗		|NO❗
└	accumulativeDividendOf	External ❗		|NO❗
DividendPayingToken	Implementation	ERC20, DividendPayingTokenInterface, Ownable		
└	<Constructor>	Public ❗	🛑	ERC20
└	<Receive Ether>	External ❗	💵	NO❗
└	distributeDividends	Public ❗	💵	NO❗
└	_withdrawDividendOfUser	Internal 🔒	🛑	
└	setRewardToken	External ❗	🛑	onlyOwner
└	swapBnbForCustomToken	Internal 🔒	🛑	
└	dividendOf	Public ❗		|NO❗
└	withdrawableDividendOf	Public ❗		|NO❗
└	withdrawnDividendOf	Public ❗		|NO❗
└	accumulativeDividendOf	Public ❗		|NO❗
└	_transfer	Internal 🔒	🛑	
└	_tokengeneration	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_setBalance	Internal 🔒	🛑	
IterableMapping	Library			
└	get	Internal 🔒		
└	getIndexOfKey	Internal 🔒		
└	getKeyAtIndex	Internal 🔒		
└	size	Internal 🔒		
└	set	Internal 🔒	🛑	
└	remove	Internal 🔒	🛑	
Address	Library			
└	sendValue	Internal 🔒	🛑	
MrsFloki	Implementation	ERC20, Ownable		
└	<Constructor>	Public ❗	🛑	ERC20
└	<Receive Ether>	External ❗	💵	NO❗
└	updateDividendTracker	Public ❗	🛑	onlyOwner
└	processDividendTracker	External ❗	🛑	NO❗
└	claim	External ❗	🛑	NO❗
└	rescueBEP20Tokens	External ❗	🛑	onlyOwner
└	rescueBNB	External ❗	🛑	NO❗
└	excludeFromFees	Public ❗	🛑	onlyOwner
└	excludeMultipleAccountsFromFees	Public ❗	🛑	onlyOwner

CONTRACT ASSESMENT

└	excludeFromDividends	External ❗	🛑	onlyOwner
└	setMarketingWallet	External ❗	🛑	onlyOwner
└	setSwapTokensAtAmount	External ❗	🛑	onlyOwner
└	setBuyTaxes	External ❗	🛑	onlyOwner
└	setSellTaxes	External ❗	🛑	onlyOwner
└	setSwapEnabled	External ❗	🛑	onlyOwner
└	enableTradingEnabled	External ❗	🛑	onlyOwner
└	setAntiBotBlocks	External ❗	🛑	onlyOwner
└	setMinBalanceForDividends	External ❗	🛑	onlyOwner
└	_setAutomatedMarketMakerPair	Private 🔐	🛑	
└	setGasForProcessing	External ❗	🛑	onlyOwner
└	setClaimWait	External ❗	🛑	onlyOwner
└	getClaimWait	External ❗		NO❗
└	getTotalDividendsDistributed	External ❗		NO❗
└	isExcludedFromFees	Public ❗		NO❗
└	withdrawableDividendOf	Public ❗		NO❗
└	getCurrentRewardToken	External ❗		NO❗
└	dividendTokenBalanceOf	Public ❗		NO❗
└	getAccountDividendsInfo	External ❗		NO❗
└	getAccountDividendsInfoAtIndex	External ❗		NO❗
└	getLastProcessedIndex	External ❗		NO❗
└	getNumberOfDividendTokenHolders	External ❗		NO❗
└	_transfer	Internal 🔒	🛑	
└	swapAndLiquify	Private 🔐	🛑	
└	swapTokensForBNB	Private 🔐	🛑	
└	addLiquidity	Private 🔐	🛑	
MrsFloki_DividendTracker	Implementation	Ownable, DividendPayingToken		
└	<Constructor>	Public ❗	🛑	DividendPayingToken
└	_transfer	Internal 🔒		
└	setMinBalanceForDividends	External ❗	🛑	onlyOwner
└	excludeFromDividends	External ❗	🛑	onlyOwner
└	updateClaimWait	External ❗	🛑	onlyOwner
└	getLastProcessedIndex	External ❗		NO❗
└	getNumberOfTokenHolders	External ❗		NO❗
└	getCurrentRewardToken	External ❗		NO❗
└	getAccount	Public ❗		NO❗
└	getAccountAtIndex	Public ❗		NO❗
└	canAutoClaim	Private 🔐		
└	setBalance	Public ❗	🛑	onlyOwner
└	process	Public ❗	🛑	NO❗

CONTRACT ASSESMENT

| └ | processAccount | Public ❗ | 🛑 | onlyOwner |

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

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-0.8.19 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in DividendPayingToken._withdrawDividendOfUser(address) (contracts/Token.sol#884-925):
        - (secondSuccess) = user.call{gas: 3000,value: _withdrawableDividend}() (contracts/Token.sol#899-902)
        - (success) = user.call{gas: 3000,value: _withdrawableDividend}() (contracts/Token.sol#911-914)
Low level call in Address.sendValue(address,uint256) (contracts/Token.sol#1121-1132):
        - (success) = recipient.call{value: amount}() (contracts/Token.sol#1127)
Low level call in MrsFloki.swapAndLiquify(uint256,uint256) (contracts/Token.sol#1564-1595):
        - (success) = address(dividendTracker).call{value: dividends}() (contracts/Token.sol#1590-1592)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function IRouter.WETH() (contracts/Token.sol#740) is not in mixedCase
Parameter DividendPayingToken.dividendOf(address)._owner (contracts/Token.sol#956) is not in mixedCase
Parameter DividendPayingToken.withdrawableDividendOf(address)._owner (contracts/Token.sol#964) is not in mixedCase
Parameter DividendPayingToken.withdrawnDividendOf(address)._owner (contracts/Token.sol#973) is not in mixedCase
Parameter DividendPayingToken.accumulativeDividendOf(address)._owner (contracts/Token.sol#984) is not in mixedCase
Constant DividendPayingToken.magnitude (contracts/Token.sol#830) is not in UPPER_CASE_WITH_UNDERSCORES
Parameter MrsFloki.setBuyTaxes(uint256,uint256,uint256)._rewards (contracts/Token.sol#1320) is not in mixedCase
Parameter MrsFloki.setBuyTaxes(uint256,uint256,uint256)._marketing (contracts/Token.sol#1321) is not in mixedCase
Parameter MrsFloki.setBuyTaxes(uint256,uint256,uint256)._liquidity (contracts/Token.sol#1322) is not in mixedCase
Parameter MrsFloki.setSellTaxes(uint256,uint256,uint256)._rewards (contracts/Token.sol#1332) is not in mixedCase
Parameter MrsFloki.setSellTaxes(uint256,uint256,uint256)._marketing (contracts/Token.sol#1333) is not in mixedCase
Parameter MrsFloki.setSellTaxes(uint256,uint256,uint256)._liquidity (contracts/Token.sol#1334) is not in mixedCase
Parameter MrsFloki.setSwapEnabled(bool)._enabled (contracts/Token.sol#1343) is not in mixedCase
Constant MrsFloki.deadWallet (contracts/Token.sol#1148-1149) is not in UPPER_CASE_WITH_UNDERSCORES
Contract MrsFloki_DividendTracker (contracts/Token.sol#1629-1882) is not in CapWords
Parameter MrsFloki_DividendTracker.getAccount(address)._account (contracts/Token.sol#1713) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (contracts/Token.sol#15)" inContext (contracts/Token.sol#9-18)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

Variable DividendPayingToken._withdrawDividendOfUser(address)._withdrawableDividend (contracts/Token.sol#887) is too similar to MrsFloki_DividendTracker.getAccount(address).withdr
awableDividends (contracts/Token.sol#1721)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

MrsFloki.setGasForProcessing(uint256) (contracts/Token.sol#1380-1391) uses literals with too many digits:
        - require(bool,string)(newValue >= 200000 && newValue <= 500000,GasForProcessing must be between 200,000 and 500,000) (contracts/Token.sol#1381-1384)
MrsFloki.slitherConstructorVariables() (contracts/Token.sol#1135-1627) uses literals with too many digits:
        - gasForProcessing = 300000 (contracts/Token.sol#1165)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

SafeMathInt.MAX_INT256 (contracts/Token.sol#594) is never used in SafeMathInt (contracts/Token.sol#592-649)
MrsFloki.currentRewardToken (contracts/Token.sol#1154) is never used in MrsFloki (contracts/Token.sol#1135-1627)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

MrsFloki.currentRewardToken (contracts/Token.sol#1154) should be constant
MrsFloki.launchtax (contracts/Token.sol#1168) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

DividendPayingToken.router (contracts/Token.sol#832) should be immutable
MrsFloki.pair (contracts/Token.sol#1139) should be immutable
MrsFloki.router (contracts/Token.sol#1138) 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/0x62dbbdb268d0335fd13e1a0ecbdc6a99321b887bc3cad7431401073638e41d36

2- Buying when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x4e5f07612fe7d835d25658400a1032babb2ebd425ac2f027b6a1d1b3399274bf

3- Selling when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x3b9767e72260bab4c6c637734f02a7259f2951dd403d017e0ee746ae20fd809a

4- Transferring when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x0050114501c083fad6fbd07169414b0360477468bc4959d7af2da47dce867cb2

5- Buying when not excluded (upto 12% tax) (passed):
https://testnet.bscscan.com/tx/0xf665941951988afb0969aecfc3e5f3c20abb7a403c444541d6fc40a7367d2da

6- Selling when not excluded (upto 12% tax) (passed):
https://testnet.bscscan.com/tx/0xb6b961d6abffa494316a9ee0c53225f88e518191283c2d01f3e485b939e8d7a2

FUNCTIONAL TESTING

7- Transferring when not excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x363659da8faee224c833a31a831a77eb0a82f679d82b2119be567e41ef3eb1ac

8- Internal swap (passed):
Marketing wallet received BNB
https://testnet.bscscan.com/address/0xa2c0c1887972f6c62a1d12326f5f308ded7ffe46#internaltx

9- Reflections (passed):
https://testnet.bscscan.com/tx/0xb6b961d6abffa494316a9ee0c53225f88e518191283c2d01f3e485b939e8d7a2

10- Auto liquidity (passed):
https://testnet.bscscan.com/tx/0xb6b961d6abffa494316a9ee0c53225f88e518191283c2d01f3e485b939e8d7a2

MANUAL TESTING

Centralization - Owner Must Enable Trades

Severity: High / Informational

Function: enableTradingEnabled

Lines: 1157

Status: Not Resolved

Overview:

The owner is required to enable trading for investors. If trading remains disabled, token holders will not have the ability to buy, sell, or transfer their tokens.

```
function enableTradingEnabled() external onlyOwner {
    require(!tradingEnabled, "Trading is already enabled");
    tradingEnabled = true;
    startTradingBlock = block.number;
}
```

Recommendation:

While the presence of this function is considered a feature rather than a flaw, it is crucial to highlight the centralization risk it inherently poses. To address this issue and ensure the enablement of trades, one possible solution would be to transfer the contract's ownership to a trusted third party, such as a Pinksale Safu developer. This would help mitigate the centralization risk associated with this function.

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