Techrate1 Techrate

TechRate

AUDIT COMPANY

Smart Contract Security Audit

TechRate

June, 2021

Audit Details

Audited project

MoonAMI

Deployer address

0x9dF22Dcfc3eb63821696A156CBd4C3F0cD5b035e

Client contacts:

MoonAMI team

Blockchain

Binance Smart Chain

Project website:

Not provided by MoonAMI team

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full.

DISCLAIMER: By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and TechRate and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (TechRate) owe no duty of care towards you or any other person, nor does TechRate make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and TechRate hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, TechRate hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against TechRate, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report.

The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

Background

TechRate was commissioned by MoonAMI to perform an audit of smart contracts:

https://bscscan.com/address/0x3a3c42dc70a73f7561166d46c708349dbde5198b#code

The purpose of the audit was to achieve the following:

- Ensure that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

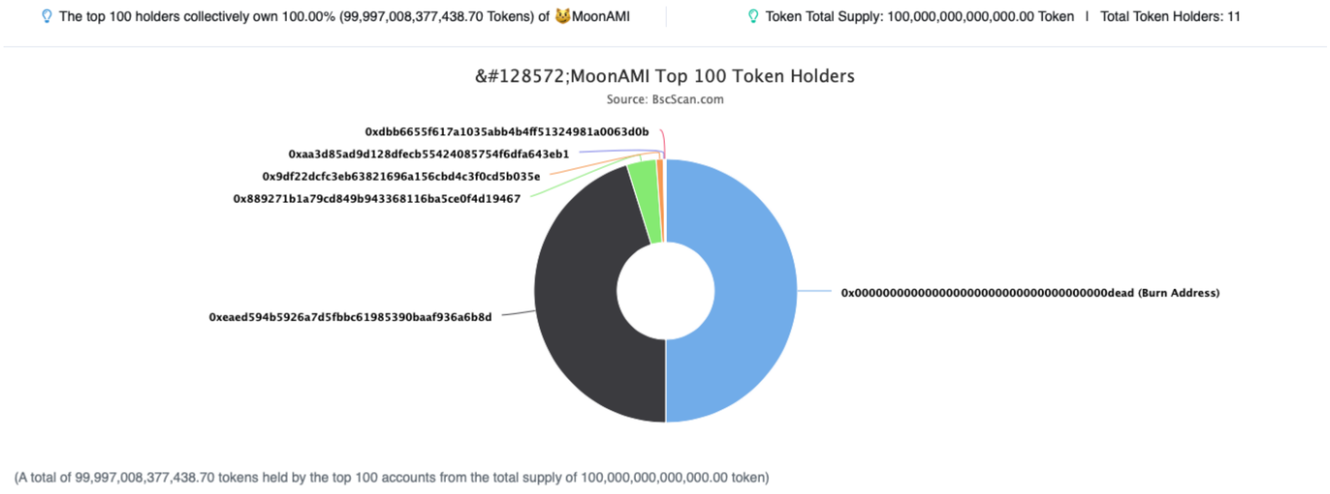
The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

Contracts Details

Token contract details for 14.06.2021

Contract name	MoonAMI
Contract address	0x3a3c42dC70A73f7561166D46C708349DbDe5198B
Total supply	100,000,000,000,000
Token ticker	MAMI
Decimals	9
Token holders	11
Transactions count	17
Top 100 holders dominance	100.00%
Total fees	5212773078295822016
Uniswap V2 pair	0xade2f00559fbb724d8e9bcf44d1232d4e6e52d8f
Contract deployer address	0x9dF22Dcfc3eb63821696A156CBd4C3F0cD5b035e
Contract's current owner address	0x9df22dcfc3eb63821696a156cbd4c3f0cd5b035e

MoonAMI Token Distribution



The top 100 holders collectively own 100.00% (99,997,008,377,438.70 Tokens) of 🐱MoonAMI

Token Total Supply: 100,000,000,000,000.00 Token | Total Token Holders: 11

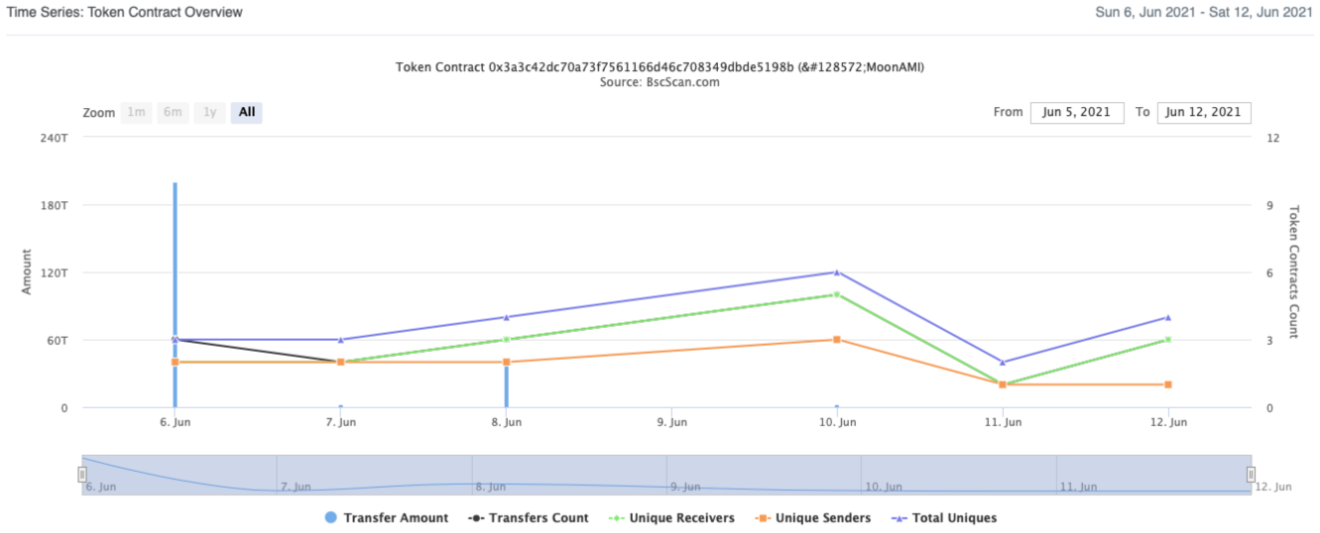
😼MoonAMI Top 100 Token Holders

Source: BscScan.com

0xdbb6655f617a1035abb4b4ff51324981a0063d0b
0xaa3d85ad9d128dfecb55424085754f6dfa643eb1
0x9df22dcfc3eb63821696a156cbd4c3f0cd5b035e
0x889271b1a79cd849b943368116ba5ce0f4d19467
0xeaed594b5926a7d5fbbc61985390baaf936a6b8d
0x000000000000000000000000000000000000dead (Burn Address)

(A total of 99,997,008,377,438.70 tokens held by the top 100 accounts from the total supply of 100,000,000,000,000.00 token)

MoonAMI Contract Interaction Details



Time Series: Token Contract Overview

Sun 6, Jun 2021 - Sat 12, Jun 2021

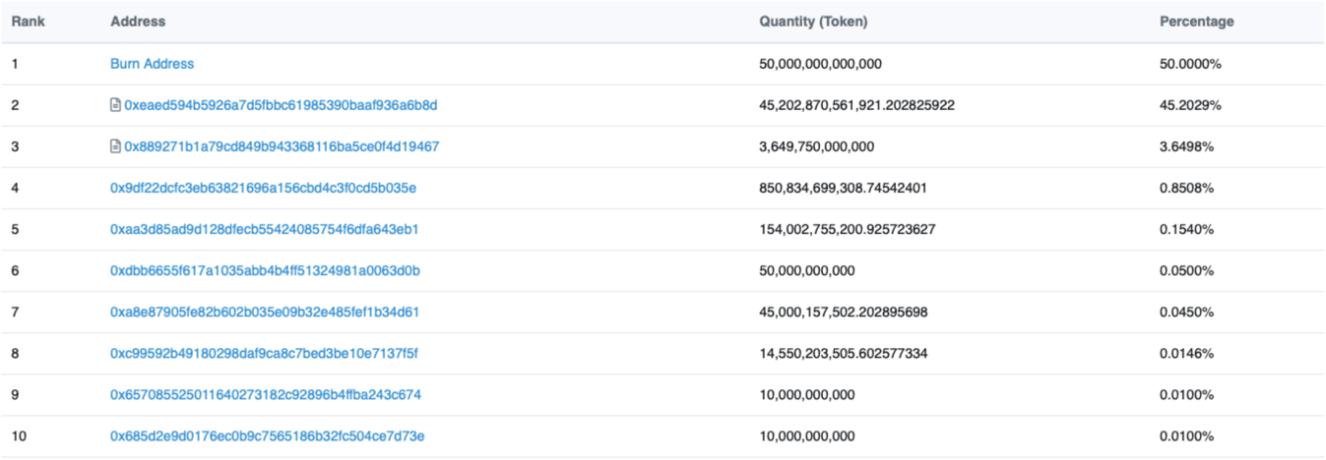
Token Contract 0x3a3c42dc70a73f7561166d46c708349dbde5198b (&#128572;MoonAMI)

Source: BscScan.com

Zoom 1m 6m 1y All

From Jun 5, 2021 To Jun 12, 2021

MoonAMI Top 10 Token Holders



Rank	Address	Quantity (Token)	Percentage
1	Burn Address	50,000,000,000,000	50.0000%
2	0xeaed594b5926a7d5fbbc61985390baaf936a6b8d	45,202,870,561,921.202825922	45.2029%
3	0x889271b1a79cd849b943368116ba5ce0f4d19467	3,649,750,000,000	3.6498%
4	0x9df22dcfc3eb63821696a156cbd4c3f0cd5b035e	850,834,699,308.74542401	0.8508%
5	0xaa3d85ad9d128dfecb55424085754f6dfa643eb1	154,002,755,200.925723627	0.1540%
6	0xdbb6655f617a1035abb4b4ff51324981a0063d0b	50,000,000,000	0.0500%
7	0xa8e87905fe82b602b035e09b32e485fef1b34d61	45,000,157,502.202895698	0.0450%
8	0xc99592b49180298daf9ca8c7bed3be10e7137f5f	14,550,203,505.602577334	0.0146%
9	0x6570855250116402731 82c92896b4ffba243c674	10,000,000,000	0.0100%
10	0x685d2e9d0176ec0b9c7565186b32fc504ce7d73e	10,000,000,000	0.0100%

Contract functions details

+ Context
  - [Int] _msgSender
  - [Int] _msgData

+ [Int] IERC20
  - [Ext] totalSupply
  - [Ext] balanceOf
  - [Ext] transfer #
  - [Ext] allowance
  - [Ext] approve #
  - [Ext] transferFrom #

+ [Lib] SafeMath
  - [Int] add
  - [Int] sub
  - [Int] sub
  - [Int] mul
  - [Int] div
  - [Int] div
  - [Int] mod
  - [Int] mod

+ [Lib] Address
  - [Int] isContract
  - [Int] sendValue #
  - [Int] functionCall #
  - [Int] functionCall #
  - [Int] functionCallWithValue #
  - [Int] functionCallWithValue #
  - [Prv] _functionCallWithValue #

+ Ownable (Context)
  - [Int] <Constructor> #
  - [Pub] owner
  - [Pub] renounceOwnership #
    - modifiers: onlyOwner
  - [Pub] transferOwnership #
    - modifiers: onlyOwner
  - [Pub] geUnlockTime
  - [Pub] lock #
    - modifiers: onlyOwner
  - [Pub] unlock #

+ [Int] IUniswapV2Factory
  - [Ext] feeTo
  - [Ext] feeToSetter
  - [Ext] getPair
  - [Ext] allPairs
  - [Ext] allPairsLength
  - [Ext] createPair #
  - [Ext] setFeeTo #

  - [Ext] setFeeToSetter #

+ [Int] IUniswapV2Pair
  - [Ext] name
  - [Ext] symbol
  - [Ext] decimals
  - [Ext] totalSupply
  - [Ext] balanceOf
  - [Ext] allowance
  - [Ext] approve #
  - [Ext] transfer #
  - [Ext] transferFrom #
  - [Ext] DOMAIN_SEPARATOR
  - [Ext] PERMIT_TYPEHASH
  - [Ext] nonces
  - [Ext] permit #
  - [Ext] MINIMUM_LIQUIDITY
  - [Ext] factory
  - [Ext] token0
  - [Ext] token1
  - [Ext] getReserves
  - [Ext] price0CumulativeLast
  - [Ext] price1CumulativeLast
  - [Ext] kLast
  - [Ext] mint #
  - [Ext] burn #
  - [Ext] swap #
  - [Ext] skim #
  - [Ext] sync #
  - [Ext] initialize #

+ [Int] IUniswapV2Router01
  - [Ext] factory
  - [Ext] WETH
  - [Ext] addLiquidity #
  - [Ext] addLiquidityETH ($)
  - [Ext] removeLiquidity #
  - [Ext] removeLiquidityETH #
  - [Ext] removeLiquidityWithPermit #
  - [Ext] removeLiquidityETHWithPermit #
  - [Ext] swapExactTokensForTokens #
  - [Ext] swapTokensForExactTokens #
  - [Ext] swapExactETHForTokens ($)
  - [Ext] swapTokensForExactETH #
  - [Ext] swapExactTokensForETH #
  - [Ext] swapETHForExactTokens ($)
  - [Ext] quote
  - [Ext] getAmountOut
  - [Ext] getAmountIn
  - [Ext] getAmountsOut
  - [Ext] getAmountsIn

+ [Int] IUniswapV2Router02 (IUniswapV2Router01)
  - [Ext] removeLiquidityETHSupportingFeeOnTransferTokens #
  - [Ext] removeLiquidityETHWithPermitSupportingFeeOnTransferTokens #

  - [Ext] swapExactTokensForTokensSupportingFeeOnTransferTokens #
  - [Ext] swapExactETHForTokensSupportingFeeOnTransferTokens ($)
  - [Ext] swapExactTokensForETHSupportingFeeOnTransferTokens #

+ MAMI (Context, IERC20, Ownable)
  - [Pub] <Constructor> #
  - [Pub] name
  - [Pub] symbol
  - [Pub] decimals
  - [Pub] totalSupply
  - [Pub] balanceOf
  - [Pub] transfer #
  - [Pub] allowance
  - [Pub] approve #
  - [Pub] transferFrom #
  - [Pub] increaseAllowance #
  - [Pub] decreaseAllowance #
  - [Pub] isExcludedFromReward
  - [Ext] setExcludeFromFee #
    - modifiers: onlyOwner
  - [Pub] totalFees
  - [Pub] deliver #
  - [Pub] reflectionFromToken
  - [Pub] tokenFromReflection
  - [Ext] excludeFromReward #
    - modifiers: onlyOwner
  - [Ext] includeInReward #
    - modifiers: onlyOwner
  - [Ext] addBotToBlackList #
    - modifiers: onlyOwner
  - [Ext] removeBotFromBlackList #
    - modifiers: onlyOwner
  - [Prv] removeAllFee #
  - [Prv] restoreAllFee #
  - [Pub] isExcludedFromFee
  - [Prv] _approve #
  - [Prv] _transfer #
  - [Prv] swapAndLiquify #
    - modifiers: lockTheSwap
  - [Prv] swapTokensForEth #
  - [Prv] addLiquidity #
  - [Prv] sendETHToCharity #
  - [Ext] manualSwap #
    - modifiers: onlyOwner
  - [Pub] manualSend #
    - modifiers: onlyOwner
  - [Ext] setSwapAndLiquifyEnabled #
    - modifiers: onlyOwner
  - [Prv] _tokenTransfer #
  - [Prv] _transferStandard #
  - [Prv] _transferToExcluded #
  - [Prv] _transferFromExcluded #
  - [Prv] _transferBothExcluded #
  - [Prv] _takeCharityLiquidity #
  - [Prv] _reflectFee #

  - [Ext] <Fallback> ($)
  - [Prv] _getValues
  - [Prv] _getTValues
  - [Prv] _getRValues
  - [Prv] _getRate
  - [Prv] _getCurrentSupply
  - [Prv] _getTaxFee
  - [Prv] _getMaxTxAmount
  - [Pub] _getETHBalance
  - [Ext] _setTaxFee #
    - modifiers: onlyOwner
  - [Ext] _setCharityFee #
    - modifiers: onlyOwner
  - [Ext] _setLiquidityFee #
    - modifiers: onlyOwner
  - [Ext] _setNumTokensSellToAddToLiquidity #
    - modifiers: onlyOwner
  - [Ext] _setMaxTxAmount #
    - modifiers: onlyOwner

($) = payable function
# = non-constant function

Issues Checking Status

	Issue description	Checking status
1.	Compiler errors.	Passed
2.	Race conditions and Reentrancy. Cross-function race conditions.	Passed
3.	Possible delays in data delivery.	Passed
4.	Oracle calls.	Passed
5.	Front running.	Passed
6.	Timestamp dependence.	Passed
7.	Integer Overflow and Underflow.	Passed
8.	DoS with Revert.	Passed
9.	DoS with block gas limit.	Low issues
10.	Methods execution permissions.	Passed
11.	Economy model of the contract.	Passed
12.	The impact of the exchange rate on the logic.	Passed
13.	Private user data leaks.	Passed
14.	Malicious Event log.	Passed
15.	Scoping and Declarations.	Passed
16.	Uninitialized storage pointers.	Passed
17.	Arithmetic accuracy.	Passed
18.	Design Logic.	Passed
19.	Cross-function race conditions.	Passed
20.	Safe Open Zeppelin contracts implementation and usage.	Passed
21.	Fallback function security.	Passed

Security Issues

High Severity Issues

No high severity issues found.

Medium Severity Issues

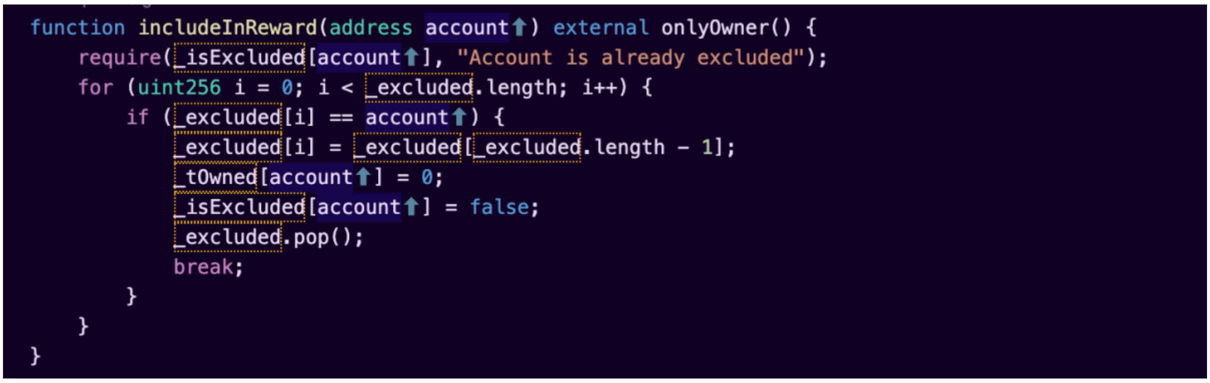
No medium severity issues found.

Low Severity Issues

1. Out of gas

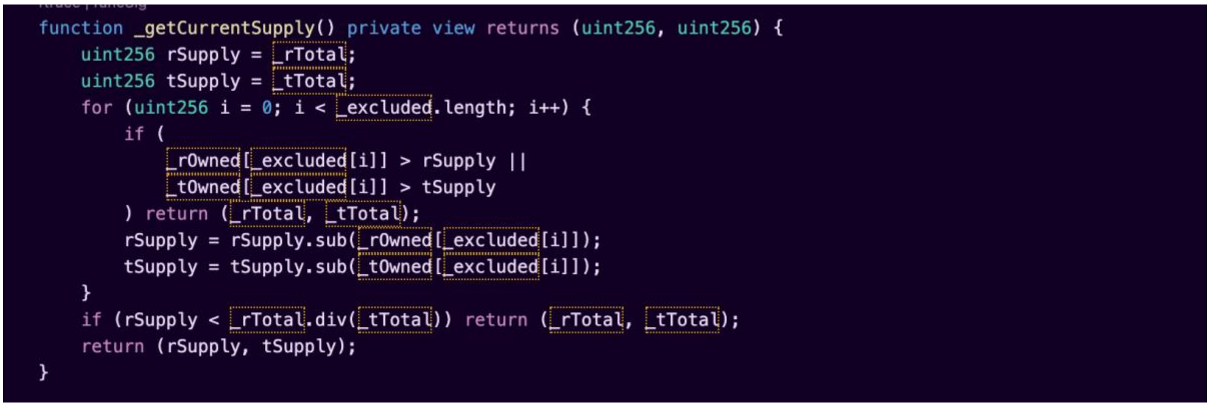
Issue:

- The function includeInReward() uses the loop to find and remove addresses from the _excluded list. Function will be aborted with OUT_OF_GAS exception if there will be a long excluded addresses list.



```
function includeInReward(address account) external onlyOwner() {
    require(_isExcluded[account], "Account is already excluded");
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (_excluded[i] == account) {
            _excluded[i] = _excluded[_excluded.length - 1];
            _tOwned[account] = 0;
            _isExcluded[account] = false;
            _excluded.pop();
            break;
        }
    }
}
```

- The function _getCurrentSupply also uses the loop for evaluating total supply. It also could be aborted with OUT_OF_GAS exception if there will be a long excluded addresses list.



```
function _getCurrentSupply() private view returns (uint256, uint256) {
    uint256 rSupply = _rTotal;
    uint256 tSupply = _tTotal;
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (
            _rOwned[_excluded[i]] > rSupply ||
            _tOwned[_excluded[i]] > tSupply
        ) return (_rTotal, _tTotal);
        rSupply = rSupply.sub(_rOwned[_excluded[i]]);
        tSupply = tSupply.sub(_tOwned[_excluded[i]]);
    }
    if (rSupply < _rTotal.div(_tTotal)) return (_rTotal, _tTotal);
    return (rSupply, tSupply);
}
```

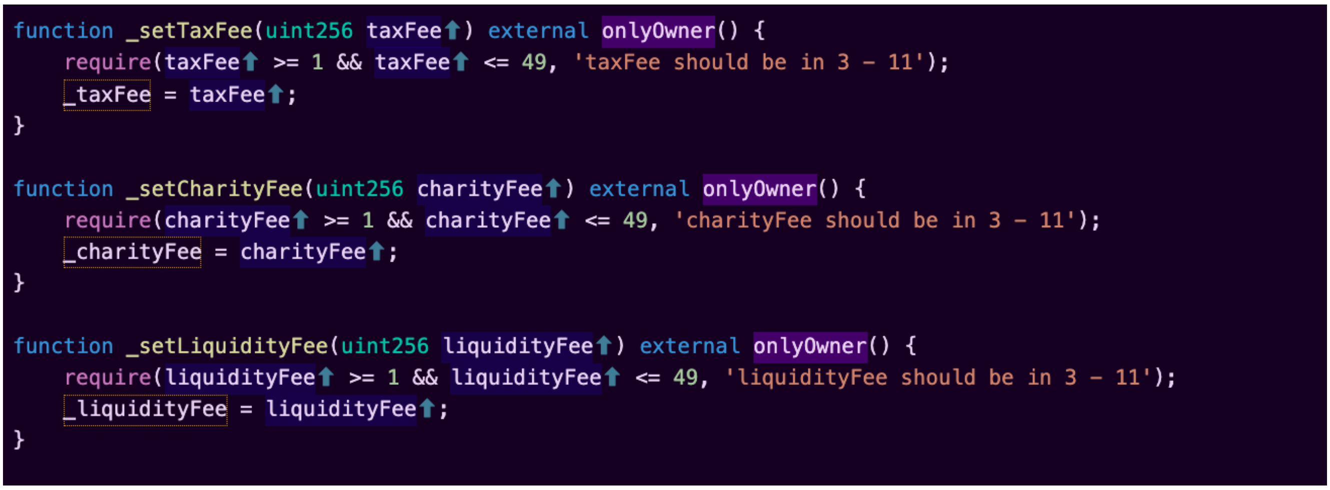
- The function removeBotFromBlackList uses the loop to remove addresses from black list array. It also could be aborted with OUT_OF_GAS exception if there will be a long addresses list.

Recommendation:

Check that the arrays length is not too big.

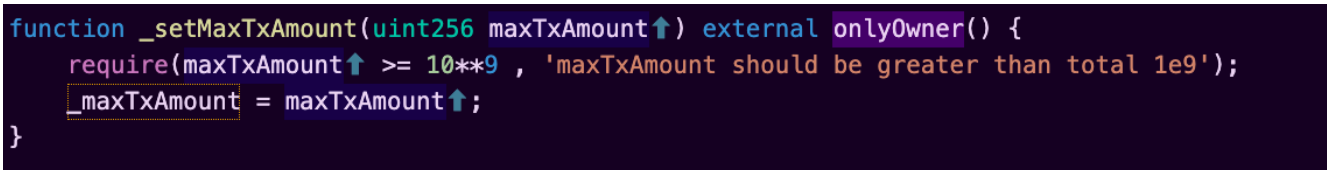
Owner privileges (In the period when the owner is not renounced)

- Owner can change the tax, charity and liquidity fee.



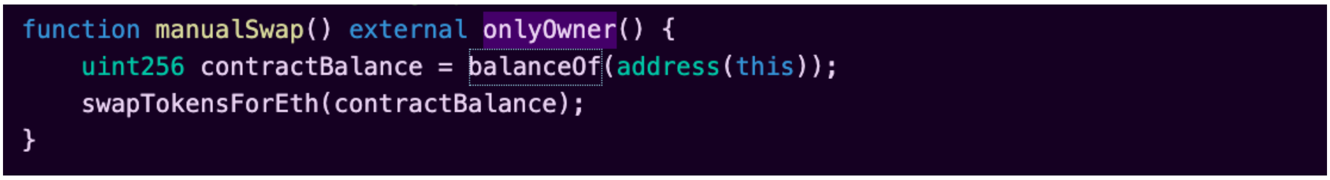
```
function _setTaxFee(uint256 taxFee) external onlyOwner() {
    require(taxFee >= 1 && taxFee <= 49, 'taxFee should be in 3 - 11');
    _taxFee = taxFee;
}

function _setCharityFee(uint256 charityFee) external onlyOwner() {
    require(charityFee >= 1 && charityFee <= 49, 'charityFee should be in 3 - 11');
    _charityFee = charityFee;
}

function _setLiquidityFee(uint256 liquidityFee) external onlyOwner() {
    require(liquidityFee >= 1 && liquidityFee <= 49, 'liquidityFee should be in 3 - 11');
    _liquidityFee = liquidityFee;
}
```

- Owner can change the maximum transaction amount.



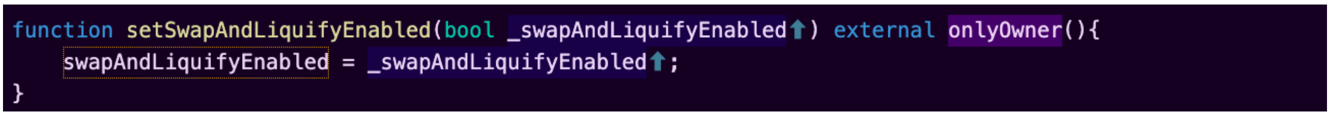
```
function _setMaxTxAmount(uint256 maxTxAmount) external onlyOwner() {
    require(maxTxAmount >= 10**9 , 'maxTxAmount should be greater than total 1e9');
    _maxTxAmount = maxTxAmount;
}
```

- Owner can manually swap tokens.



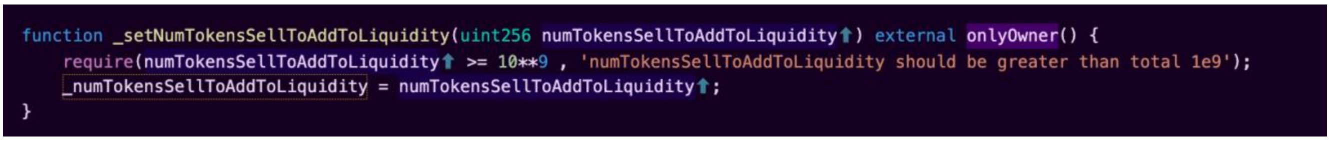
```
function manualSwap() external onlyOwner() {
    uint256 contractBalance = balanceOf(address(this));
    swapTokensForEth(contractBalance);
}
```

- Owner can disable and enable swapAndLiquify function.



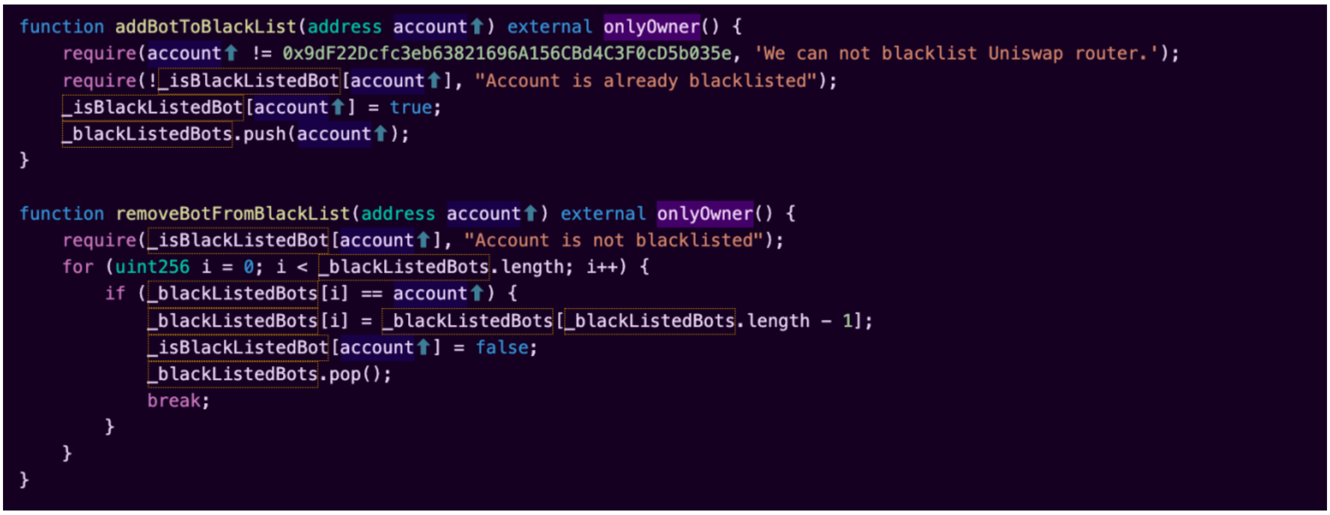
```
function setSwapAndLiquifyEnabled(bool _swapAndLiquifyEnabled) external onlyOwner(){
    swapAndLiquifyEnabled = _swapAndLiquifyEnabled;
}
```

- Owner can change minimum number of tokens to add to liquidity.



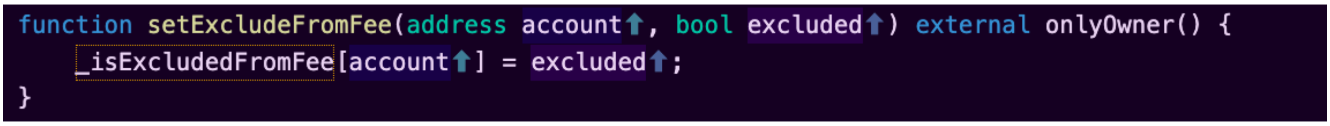
```
function _setNumTokensSellToAddToLiquidity(uint256 numTokensSellToAddToLiquidity) external onlyOwner() {
    require(numTokensSellToAddToLiquidity >= 10**9 , 'numTokensSellToAddToLiquidity should be greater than total 1e9');
    _numTokensSellToAddToLiquidity = numTokensSellToAddToLiquidity;
}
```

- Owner can include and exclude addresses in black list.



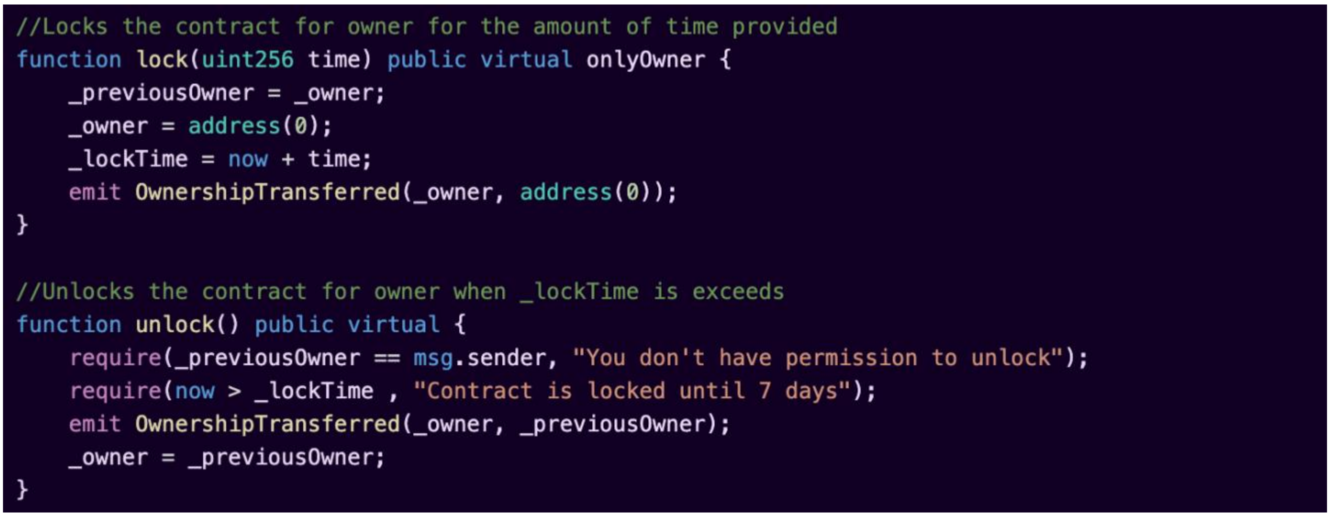
```
function addBotToBlackList(address account) external onlyOwner() {
    require(account != 0x9dF22Dcfc3eb63821696A156CBd4C3F0cD5b035e, 'We can not blacklist Uniswap router.');
    require(!_isBlackListedBot[account], "Account is already blacklisted");
    _isBlackListedBot[account] = true;
    _blackListedBots.push(account);
}

function removeBotFromBlackList(address account) external onlyOwner() {
    require(_isBlackListedBot[account], "Account is not blacklisted");
    for (uint256 i = 0; i < _blackListedBots.length; i++) {
        if (_blackListedBots[i] == account) {
            _blackListedBots[i] = _blackListedBots[_blackListedBots.length - 1];
            _isBlackListedBot[account] = false;
            _blackListedBots.pop();
            break;
        }
    }
}
```

- **Owner can exclude from the fee.**



```
function setExcludeFromFee(address account, bool excluded) external onlyOwner() {
    _isExcludedFromFee[account] = excluded;
}
```

- **Owner can lock and unlock. By the way, using these functions the owner could retake privileges even after the ownership was renounced.**



```
//Locks the contract for owner for the amount of time provided
function lock(uint256 time) public virtual onlyOwner {
    _previousOwner = _owner;
    _owner = address(0);
    _lockTime = now + time;
    emit OwnershipTransferred(_owner, address(0));
}

//Unlocks the contract for owner when _lockTime is exceeds
function unlock() public virtual {
    require(_previousOwner == msg.sender, "You don't have permission to unlock");
    require(now > _lockTime , "Contract is locked until 7 days");
    emit OwnershipTransferred(_owner, _previousOwner);
    _owner = _previousOwner;
}
```

Conclusion

Smart contracts contain low severity issues! Liquidity pair contract's security is not checked due to out of scope.

Liquidity locking details NOT provided by the team.

TechRate note:

Please check the disclaimer above and note, the audit makes no statements or warranties on business model, investment attractiveness or code sustainability. The report is provided for the only contract mentioned in the report and does not include any other potential contracts deployed by Owner.