Techrate1 Techrate

TechRate

AUDIT COMPANY

Smart Contract Security Audit

TechRate

November, 2021

Audit Details

Audited project

Transhuman Coin

Deployer address

0x018fbdf1d7085781d321e8fbb25004c3dbfa1f9a

Client contacts:

Transhuman Coin team

Blockchain

Binance Smart Chain

Project website:

www.transhumancoin.finance

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 Transhuman Coin to perform an audit of smart contracts:
https://bscscan.com/address/0x56083560594e314b5cdd1680ec6a493bb851bbd8#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 24.11.2021

Contract name	Transhuman Coin
Contract address	0x56083560594E314b5cDd1680eC6a493bb851BBd8
Total supply	7,000,000,000
Token ticker	THC
Decimals	9
Token holders	11,005
Transactions count	52,049
Top 100 holders dominance	78.47%
Liquidity fee	4
Tax fee	2
Total fees	551431717849766923
Uniswap V2 pair	0x62be1533f3a78de99ca297ebbe489a3fb7253bef2
Contract deployer address	0x018fbdf1d7085781d321e8fbb25004c3dbfa1f9a
Contract’s current owner address	0x018fbdf1d7085781d321e8fbb25004c3dbfa1f9a

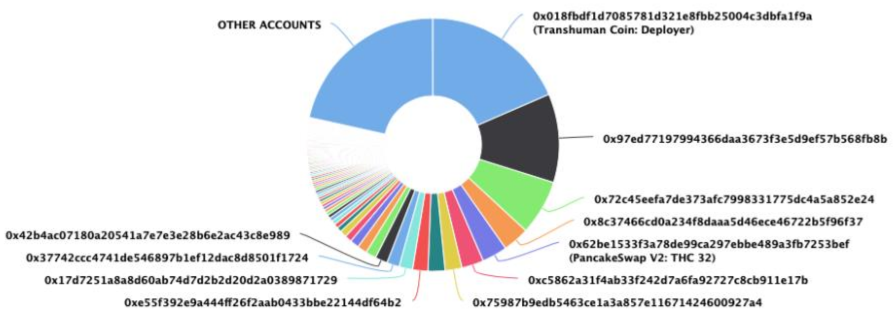
Transhuman Coin Token Distribution

The top 100 holders collectively own 78.47% (5,492,664,052.05 Tokens) of Transhuman Coin

Token Total Supply: 7,000,000,000.00 Token | Total Token Holders: 11,005

Transhuman Coin Top 100 Token Holders

Source: BscScan.com



(A total of 5,492,664,052.05 tokens held by the top 100 accounts from the total supply of 7,000,000,000.00 token)

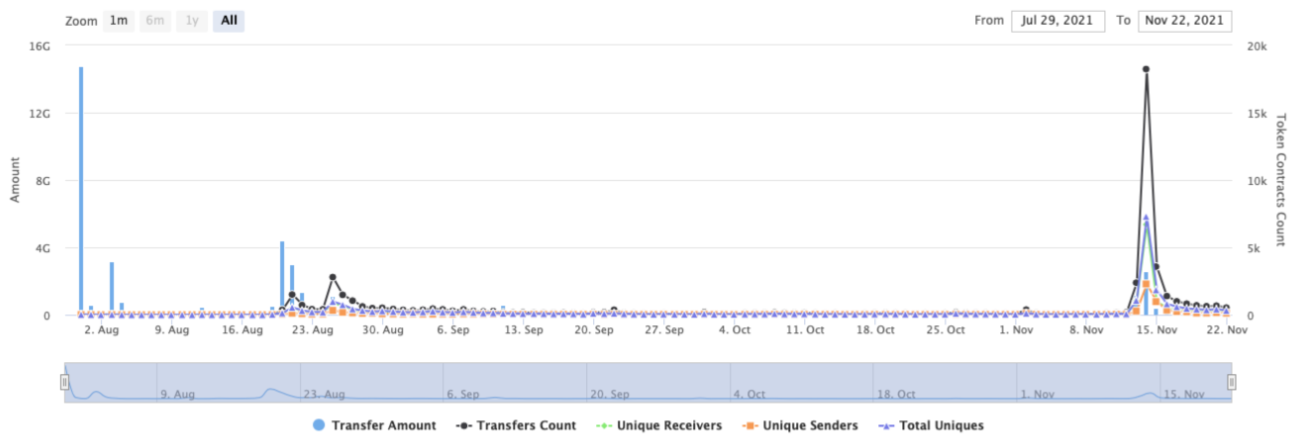
Transhuman Coin Interaction details

Time Series: Token Contract Overview

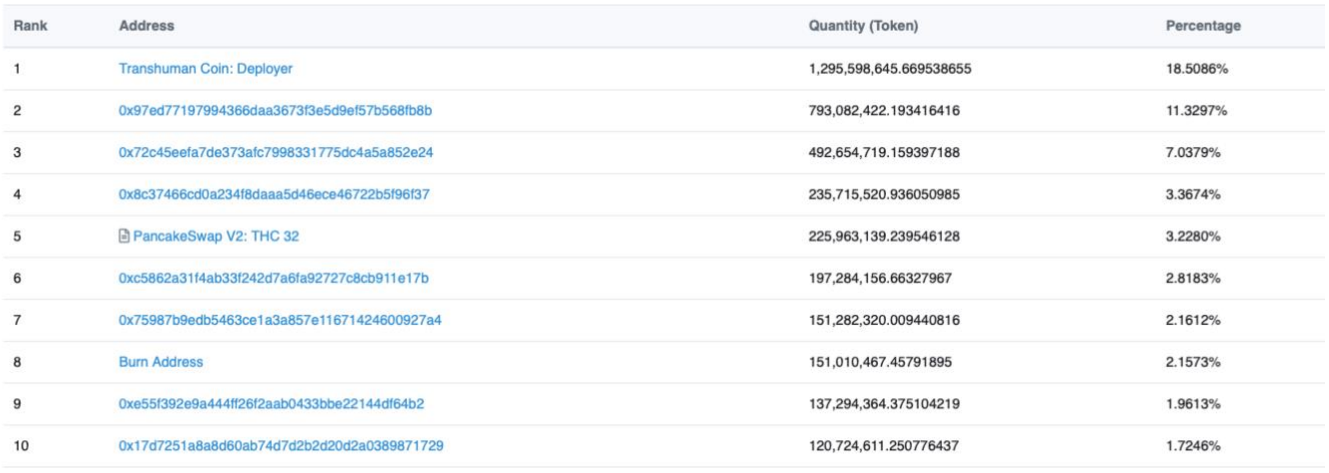
Sat 31, Jul 2021 - Mon 22, Nov 2021

Token Contract 0x56083560594e314b5cdd1680ec6a493bb851bbd8 (Transhuman Coin)

Source: BscScan.com



Transhuman Coin Top 10 Token Holders



Rank	Address	Quantity (Token)	Percentage
1	Transhuman Coin: Deployer	1,295,598,645.669538655	18.5086%
2	0x97ed77197994366daa3673f3e5d9ef57b568fb8b	793,082,422.193416416	11.3297%
3	0x72c45eefa7de373afc7998331775dc4a5a852e24	492,654,719.159397188	7.0379%
4	0x8c37466cd0a234f8daaa5d46ece46722b5f96f37	235,715,520.936050985	3.3674%
5	PancakeSwap V2: THC 32	225,963,139.239546128	3.2280%
6	0xc5862a31f4ab33f242d7a6fa92727c8cb911e17b	197,284,156.66327967	2.8183%
7	0x75987b9edb5463ce1a3a857e11671424600927a4	151,282,320.009440816	2.1612%
8	Burn Address	151,010,467.45791895	2.1573%
9	0xe55f392e9a444ff26f2aab0433bbe22144df64b2	137,294,364.375104219	1.9613%
10	0x17d7251a8a8d60ab74d7d2b2d20d2a0389871729	120,724,611.250776437	1.7246%

Contract functions details

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

+ [Int] IBEP20
  - [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

+  Ownable (Context)
  - [Pub] <Constructor> #
  - [Pub] owner
  - [Pub] renounceOwnership #
    - modifiers: onlyOwner
  - [Pub] transferOwnership #
    - modifiers: onlyOwner
  - [Pub] getUnlockTime
  - [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 #

+ TranshumanCoin (Context, IBEP20, Ownable)
  - [Pub] <Constructor> #
    - modifiers: Ownable
  - [Pub] name
  - [Pub] symbol

- [Pub] decimals
- [Pub] totalSupply
- [Pub] balanceOf
- [Pub] transfer #
- [Pub] allowance
- [Pub] approve #
- [Pub] transferFrom #
- [Pub] increaseAllowance #
- [Pub] decreaseAllowance #
- [Pub] isExcludedFromReward
- [Pub] totalFees
- [Pub] deliver #
- [Pub] reflectionFromToken
- [Pub] tokenFromReflection
- [Pub] excludeFromReward #
  - modifiers: onlyOwner
- [Ext] includeInReward #
  - modifiers: onlyOwner
- [Ext] setmarketingWallet #
  - modifiers: onlyOwner
- [Ext] setExcludedFromFee #
  - modifiers: onlyOwner
- [Ext] setTaxFeePercent #
  - modifiers: onlyOwner
- [Ext] setLiquidityFeePercent #
  - modifiers: onlyOwner
- [Ext] setPercentageOfLiquidityFormarketing #
  - modifiers: onlyOwner
- [Ext] setMaxTxAmount #
  - modifiers: onlyOwner
- [Pub] setSwapAndLiquifyEnabled #
  - modifiers: onlyOwner
- [Ext] <Fallback> ($)
- [Ext] setUniswapRouter #
  - modifiers: onlyOwner
- [Ext] setUniswapPair #
  - modifiers: onlyOwner
- [Ext] setExcludedFromAutoLiquidity #
  - modifiers: onlyOwner
- [Prv] _reflectFee #
- [Prv] _getTValues
- [Prv] _getRValues
- [Prv] _getRate
- [Prv] _getCurrentSupply
- [Prv] takeTransactionFee #
- [Prv] calculateFee
- [Pub] isExcludedFromFee
- [Prv] _approve #
- [Prv] _transfer #
- [Prv] swapAndLiquify #
  - modifiers: lockTheSwap
- [Prv] swapTokensForBnb #
- [Prv] addLiquidity #
- [Prv] _tokenTransfer #
- [Prv] _transferStandard #

   - **[Prv]** _transferBothExcluded **#**
   - **[Prv]** _transferToExcluded **#**
   - **[Prv]** _transferFromExcluded **#**

($) = 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

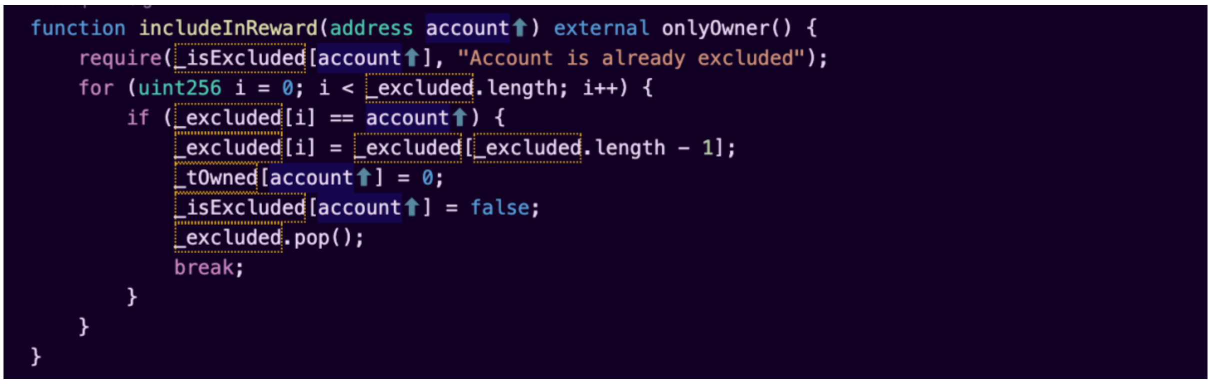
No medium severity issues found.

Low Severity Issues

1. Out of gas

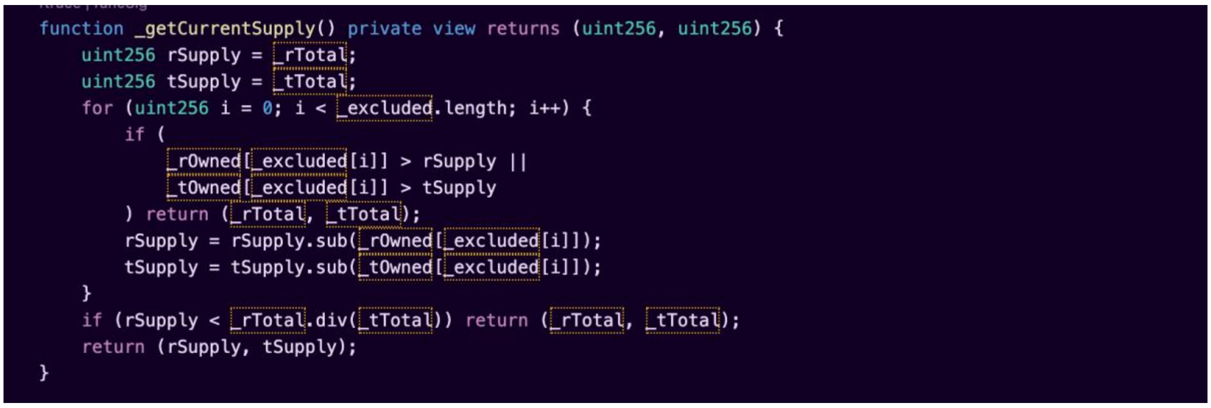
Issue:

- 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.



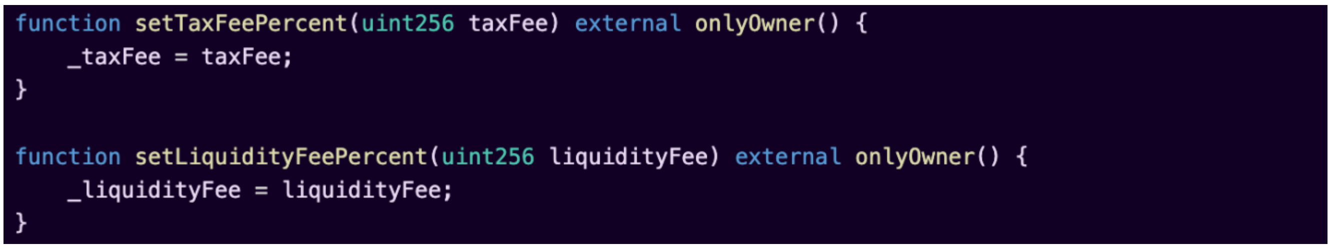
```
function _getCurrentSupply() private view returns (uint256, uint256) {
    uint256 rSupply = _rTotal;
    uint256 tSupply = _tTotal;
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (
            _rOwned[_excluded[i]] > rSupply ||
            _tOwned[_excluded[i]] > tSupply
      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

Recommendation:

Check that the excluded array length is not too big.

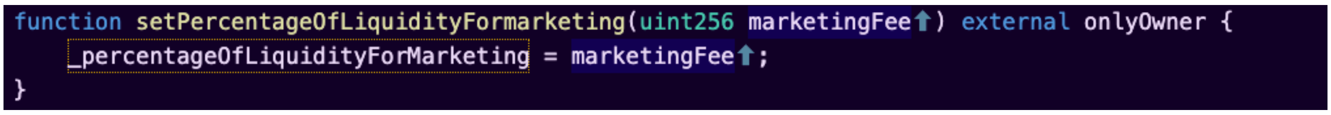
Owner privileges (In the period when the owner is not renounced)

- Owner can change the tax and liquidity fee.



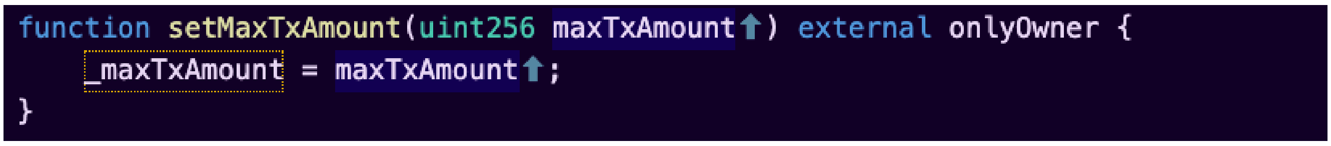
```
function setTaxFeePercent(uint256 taxFee) external onlyOwner() {
    _taxFee = taxFee;
}

function setLiquidityFeePercent(uint256 liquidityFee) external onlyOwner() {
    _liquidityFee = liquidityFee;
}
```

- Owner can change dev fee.



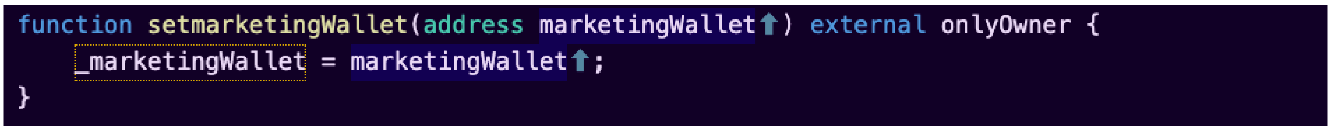
```
function setPercentageOfLiquidityFormarketing(uint256 marketingFee) external onlyOwner {
    _percentageOfLiquidityForMarketing = marketingFee;
}
```

- Owner can change the maximum transaction amount.



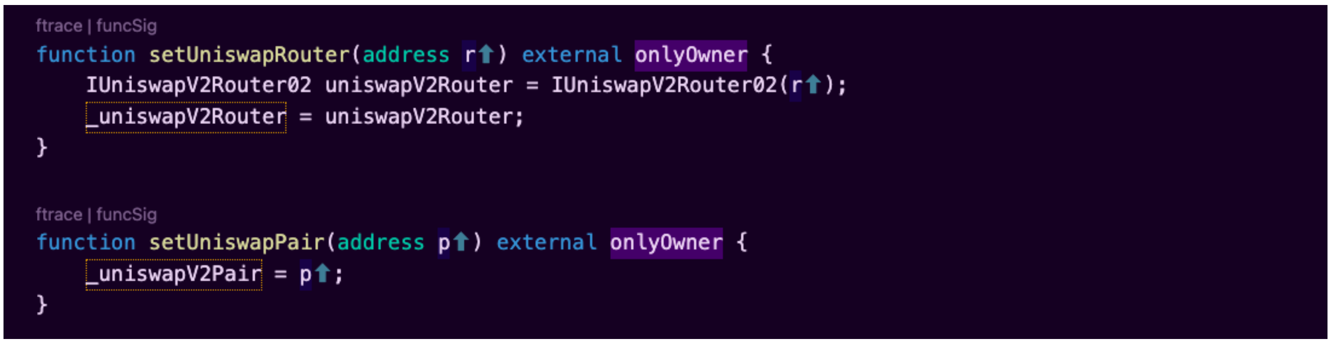
```
function setMaxTxAmount(uint256 maxTxAmount) external onlyOwner {
    _maxTxAmount = maxTxAmount;
}
```

- Owner can change dev wallet.



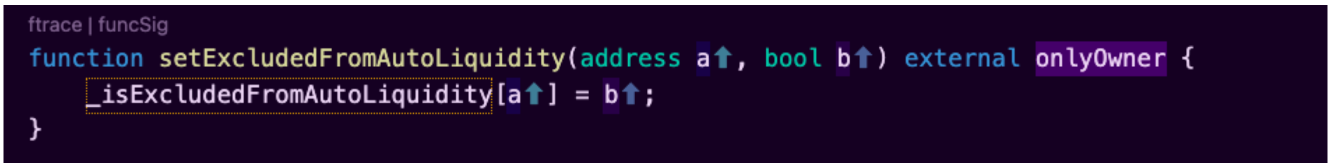
```
function setmarketingWallet(address marketingWallet) external onlyOwner {
    _marketingWallet = marketingWallet;
}
```

- Owner can change uniswap router and pair.



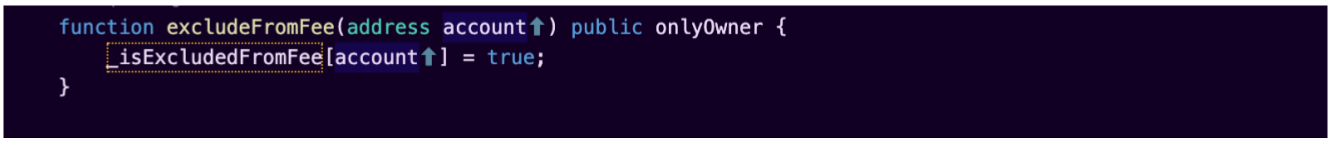
```
ftrace | funcSig
function setUniswapRouter(address r) external onlyOwner {
    IUniswapV2Router02 uniswapV2Router = IUniswapV2Router02(r);
    _uniswapV2Router = uniswapV2Router;
}

ftrace | funcSig
function setUniswapPair(address p) external onlyOwner {
    _uniswapV2Pair = p;
}
```

- Owner can exclude from and include to autoliquidity.



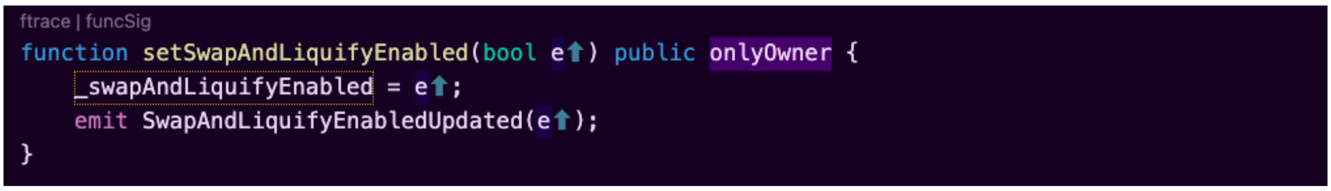
```
ftrace | funcSig
function setExcludedFromAutoLiquidity(address a, bool b) external onlyOwner {
    _isExcludedFromAutoLiquidity[a] = b;
}
```

- Owner can exclude from the fee.



```
function excludeFromFee(address account) public onlyOwner {
    _isExcludedFromFee[account] = true;
}
```

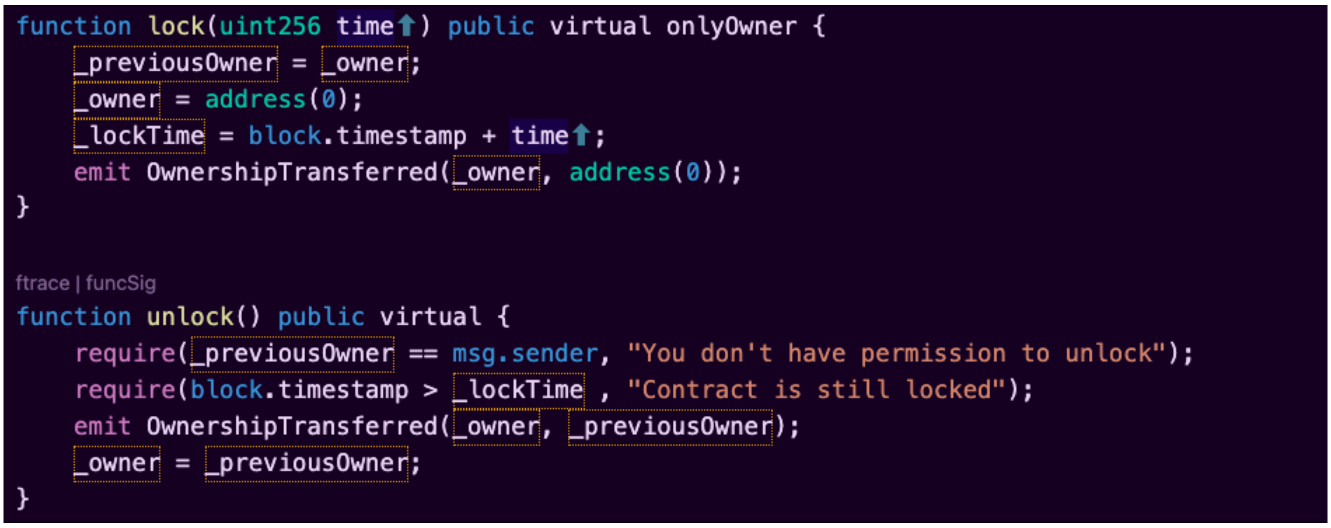
- Owner can disable and enable swap and liquify.



ftrace | funcSig

```
function setSwapAndLiquifyEnabled(bool e) public onlyOwner {
    _swapAndLiquifyEnabled = e;
    emit SwapAndLiquifyEnabledUpdated(e);
}
```

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



```
function lock(uint256 time) public virtual onlyOwner {
    _previousOwner = _owner;
    _owner = address(0);
    _lockTime = block.timestamp + time;
    emit OwnershipTransferred(_owner, address(0));
}
```

ftrace | funcSig

```
function unlock() public virtual {
    require(_previousOwner == msg.sender, "You don't have permission to unlock");
    require(block.timestamp > _lockTime , "Contract is still locked");
    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 provided by the team:
https://dxsale.app/app/v2_9/dxlockview?id=0&add=0x018fBDF1d7085781d321e8fbB25004C3dBfa1F9a&type=lplock&chain=BSC

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.