Techrate1 Techrate

TechRate

AUDIT COMPANY

# Smart Contract Security Audit

TechRate

June, 2021

# Audit Details

Audited project

**Pyro**

Deployer address

**0xb72182454963f0019c6af89da0de14879e9f14cb**

Client contacts:

**Pyro team**

Blockchain

**Binance Smart Chain**

Project website:

**[www.pyromaniac.io](www.pyromaniac.io)**

# 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 Pyro to perform an audit of smart contracts:**
https://bscscan.com/address/0x54d0a5010d09aabc1f429a159d1931007f4c7a6b#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.**

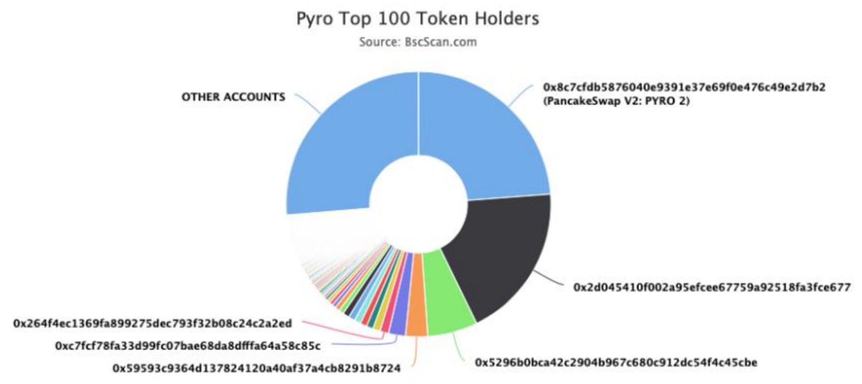
# Contracts Details

## Token contract details for 22.08.2021

| | |
|---|---|
| Contract name | Pyro |
| Contract address | 0x54D0a5010D09AaBC1f429A159d1931007f4c7a6b |
| Total supply | 1,000,000,000,000 |
| Token ticker | PYRO |
| Decimals | 9 |
| Token holders | 2,298 |
| Transactions count | 9,748 |
| Top 100 holders dominance | 73.69% |
| Liquidity fee | 6 |
| Tax fee | 4 |
| Total fees | 38739847653619732576 |
| Uniswap V2 pair | 0x8c7cfdb5876040e9391e37e69f0e476c49e2d7b2 |
| Contract deployer address | 0xb72182454963f0019c6af89da0de14879e9f14cb |
| Contract's current owner address | 0xb72182454963f0019c6af89da0de14879e9f14cb |

## Pyro Token Distribution

The top 100 holders collectively own 73.69% (736,945,487,958.20 Tokens) of Pyro | Token Total Supply: 1,000,000,000,000.00 Token | Total Token Holders: 2,298

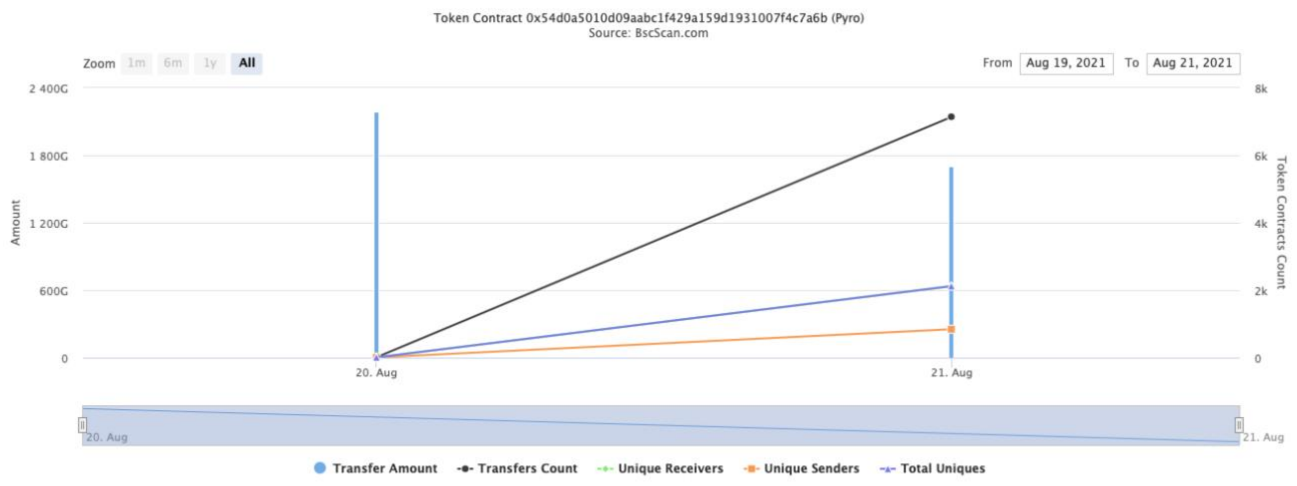


(A total of 736,945,487,958.20 tokens held by the top 100 accounts from the total supply of 1,000,000,000,000.00 token)

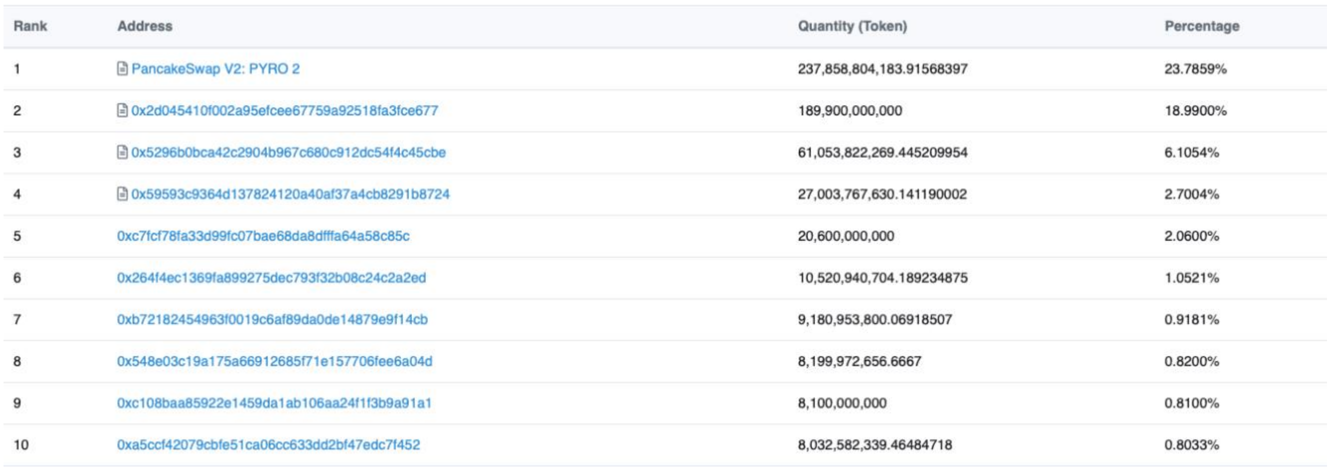
## Pyro Contract Interaction Details

Time Series: Token Contract Overview

Fri 20, Aug 2021 - Sat 21, Aug 2021



# Pyro Top 10 Token Holders



| Rank | Address | Quantity (Token) | Percentage |
|---|---|---|---|
| 1 | PancakeSwap V2: PYRO 2 | 237,858,804,183.91568397 | 23.7859% |
| 2 | 0x2d045410f002a95efcee67759a92518fa3fce677 | 189,900,000,000 | 18.9900% |
| 3 | 0x5296b0bca42c2904b967c680c912dc54f4c45cbe | 61,053,822,269.445209954 | 6.1054% |
| 4 | 0x59593c9364d137824120a40af37a4cb8291b8724 | 27,003,767,630.141190002 | 2.7004% |
| 5 | 0xc7fcf78fa33d99fc07bae68da8dfffa64a58c85c | 20,600,000,000 | 2.0600% |
| 6 | 0x264f4ec1369fa899275dec793f32b08c24c2a2ed | 10,520,940,704.189234875 | 1.0521% |
| 7 | 0xb72182454963f0019c6af89da0de14879e9f14cb | 9,180,953,800.06918507 | 0.9181% |
| 8 | 0x548e03c19a175a66912685f71e157706fee6a04d | 8,199,972,656.6667 | 0.8200% |
| 9 | 0xc108baa85922e1459da1ab106aa24f1f3b9a91a1 | 8,100,000,000 | 0.8100% |
| 10 | 0xa5ccf42079cbfe51ca06cc633dd2bf47edc7f452 | 8,032,582,339.46484718 | 0.8033% |

# 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 #

+ Pyro (Context, IBEP20, 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
- [Pub] totalFees
- [Pub] reflectionFromToken
- [Pub] tokenFromReflection
- [Pub] excludeFromReward #
  - modifiers: onlyOwner
- [Ext] includeInReward #
  - modifiers: onlyOwner
- [Ext] setExcludedFromFee #
  - modifiers: onlyOwner
- [Ext] setTaxFeePercent #
  - modifiers: onlyOwner
- [Ext] setLiquidityFeePercent #
  - modifiers: onlyOwner
- [Ext] setMaxTx #
  - modifiers: onlyOwner
- [Ext] setDevWallet #
  - modifiers: onlyOwner
- [Ext] setMarketingWallet #
  - modifiers: onlyOwner
- [Ext] setCharityWallet #
  - modifiers: onlyOwner
- [Ext] setMinTokenBalance #
  - modifiers: onlyOwner
- [Ext] setAntiWhaleEnabled #
  - modifiers: onlyOwner
- [Ext] setExcludedFromAntiWhale #
  - modifiers: onlyOwner
- [Ext] setExcludedFromBuy #
  - modifiers: onlyOwner
- [Ext] setExcludedFromMaxTx #
  - modifiers: onlyOwner
- [Ext] setPercentageOfLiquidityForDev #
  - modifiers: onlyOwner
- [Ext] setPercentageOfLiquidityForMarketing #
  - modifiers: onlyOwner
- [Ext] setAntiWhaleThreshold #
  - modifiers: onlyOwner
- [Ext] setPercentageOfLiquidityForCharity #
  - 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 #

($) = 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

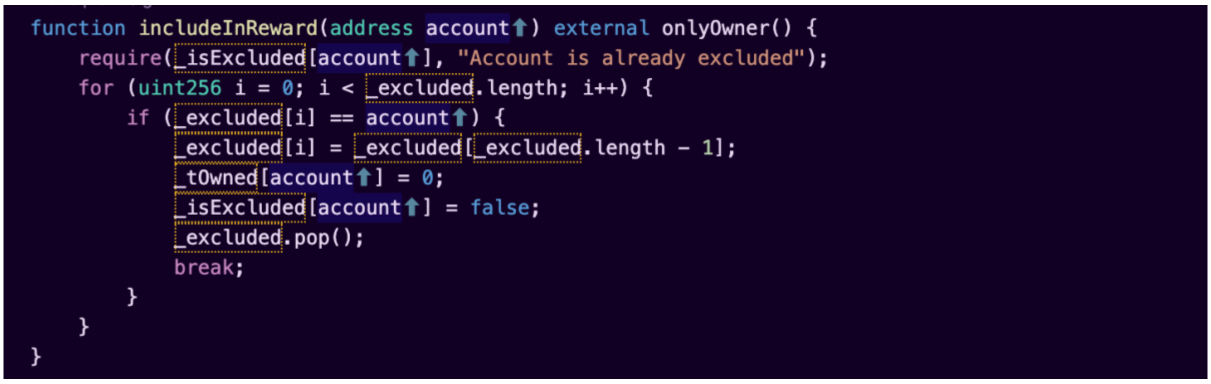
No medium severity issues found.

## Low Severity Issues

### 1. Out of gas

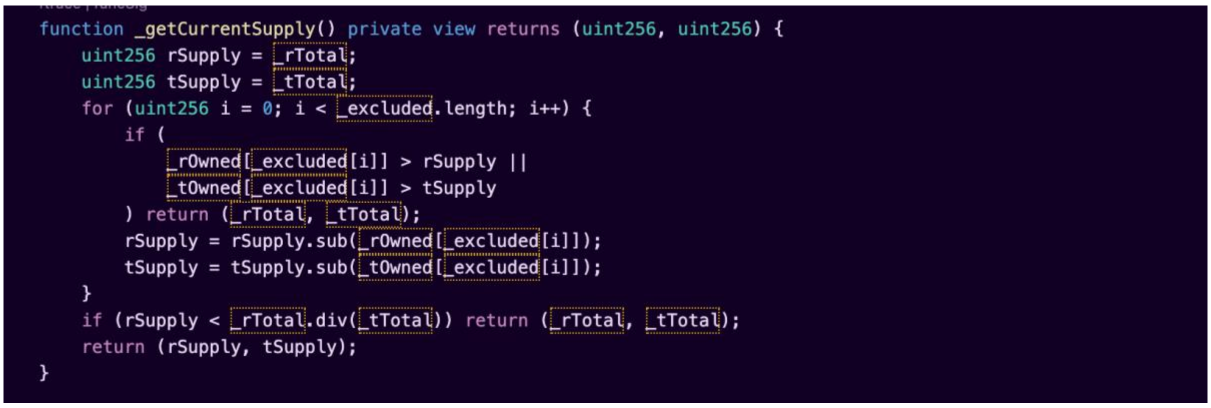
**Issue:**

- 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**:

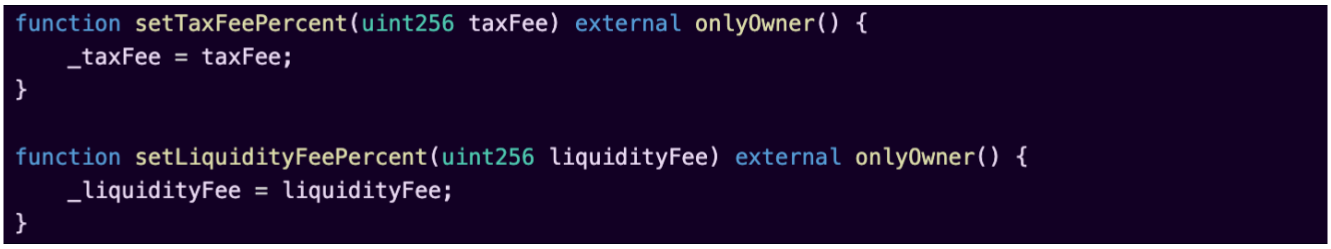
Check that the excluded array length is not too big.

## Notes:

- swapAndLiquify() function adds liquidity to smaller amount of tokens.

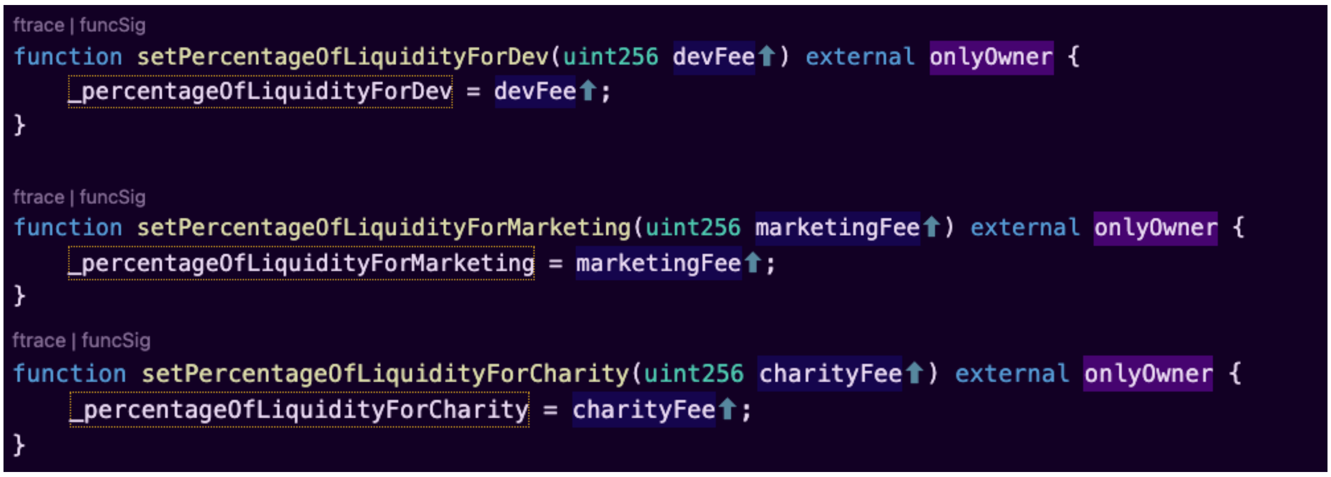
## Owner privileges (In the period when the owner is not renounced)

- Owner can change the tax and liquidity fee.



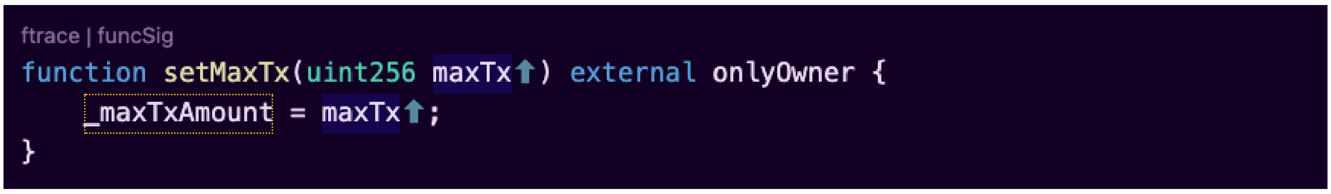
```
function setTaxFeePercent(uint256 taxFee) external onlyOwner() {
    _taxFee = taxFee;
}

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

- Owner can change liquidity percentage for dev, marketing and charity.



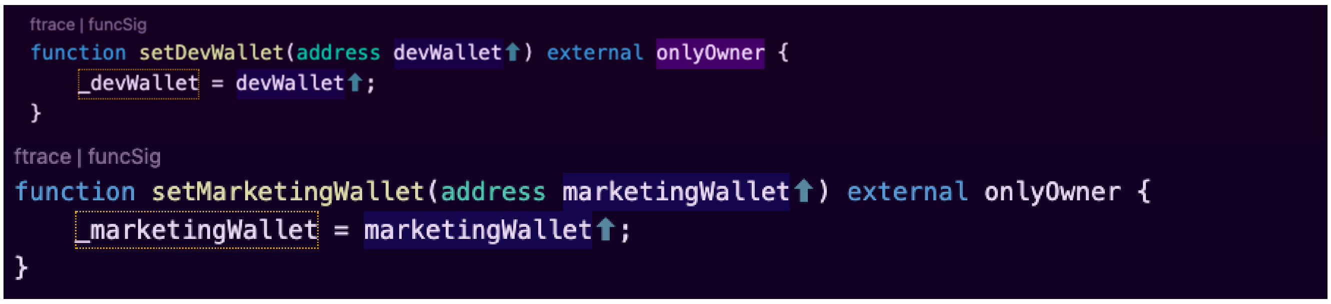
```
ftrace | funcSig
function setPercentageOfLiquidityForDev(uint256 devFee) external onlyOwner {
    _percentageOfLiquidityForDev = devFee;
}

ftrace | funcSig
function setPercentageOfLiquidityForMarketing(uint256 marketingFee) external onlyOwner {
    _percentageOfLiquidityForMarketing = marketingFee;
}
ftrace | funcSig
function setPercentageOfLiquidityForCharity(uint256 charityFee) external onlyOwner {
    _percentageOfLiquidityForCharity = charityFee;
}
```

- Owner can change the maximum transaction amount.

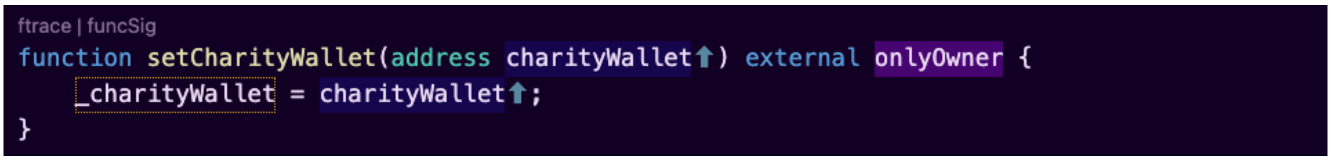


```
ftrace | funcSig
function setMaxTx(uint256 maxTx) external onlyOwner {
    _maxTxAmount = maxTx;
}
```

- Owner can change dev, charity and marketing wallet.

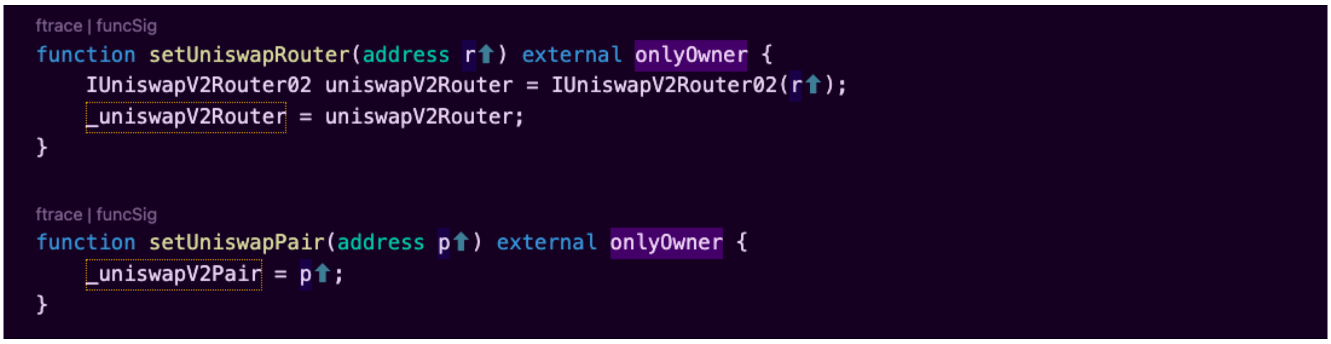


```
ftrace | funcSig
function setDevWallet(address devWallet) external onlyOwner {
    _devWallet = devWallet;
}
ftrace | funcSig
function setMarketingWallet(address marketingWallet) external onlyOwner {
    _marketingWallet = marketingWallet;
}
```



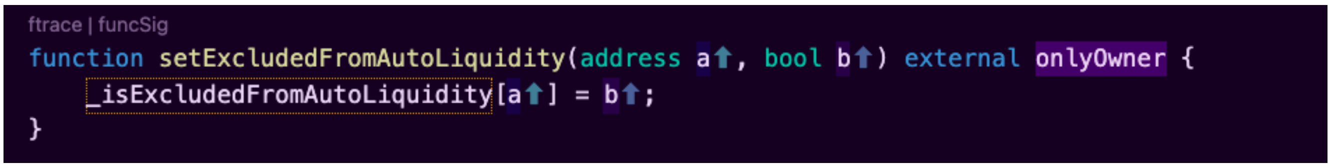
```
ftrace | funcSig
function setCharityWallet(address charityWallet) external onlyOwner {
    _charityWallet = charityWallet;
}
```

- Owner can change uniswap router and pair.



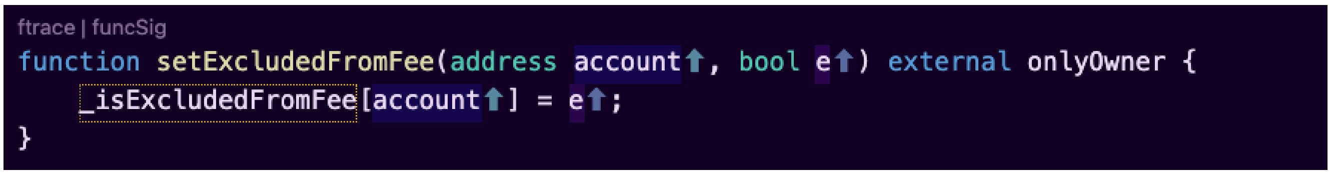
```
ftrace | funcSig
function setUniswapRouter(address r) external onlyOwner {
    IUniswapV2Router02 uniswapV2Router = IUniswapV2Router02(r);
    _uniswapV2Router = uniswapV2Router;
}

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

- Owner can exclude from and include to autoliquidity.



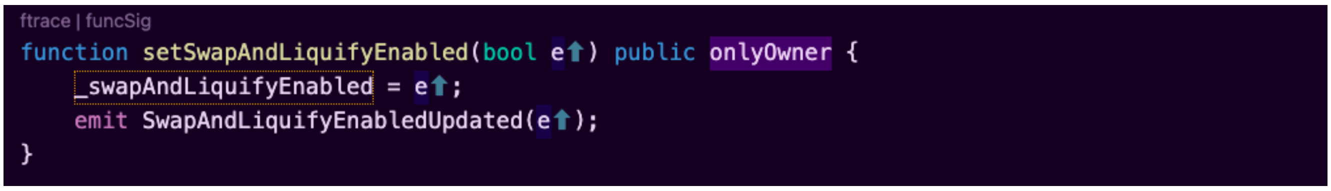
```
ftrace | funcSig
function setExcludedFromAutoLiquidity(address a, bool b) external onlyOwner {
    _isExcludedFromAutoLiquidity[a] = b;
}
```

- Owner can exclude from the fee.



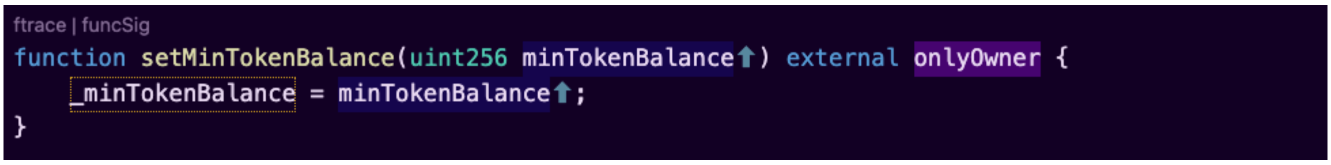
```
ftrace | funcSig
function setExcludedFromFee(address account, bool e) external onlyOwner {
    _isExcludedFromFee[account] = e;
}
```

- Owner can disable and enable swap and liquify.



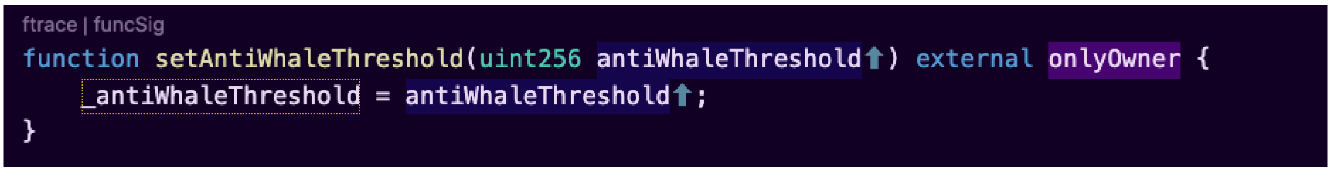
```
ftrace | funcSig
function setSwapAndLiquifyEnabled(bool e) public onlyOwner {
    _swapAndLiquifyEnabled = e;
    emit SwapAndLiquifyEnabledUpdated(e);
}
```

- Owner can change _minTokenBalance.



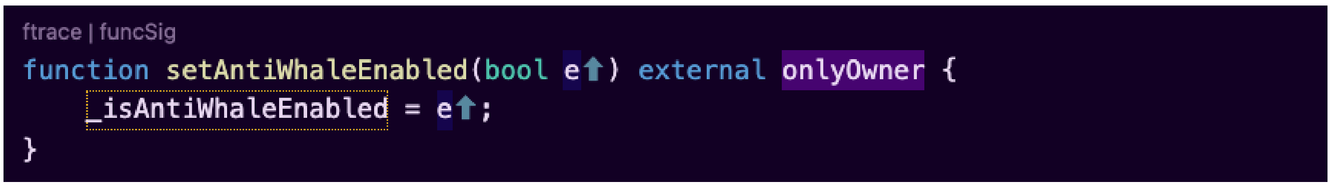
```
ftrace | funcSig
function setMinTokenBalance(uint256 minTokenBalance) external onlyOwner {
    _minTokenBalance = minTokenBalance;
}
```

- Owner can set antiWhale threshold.



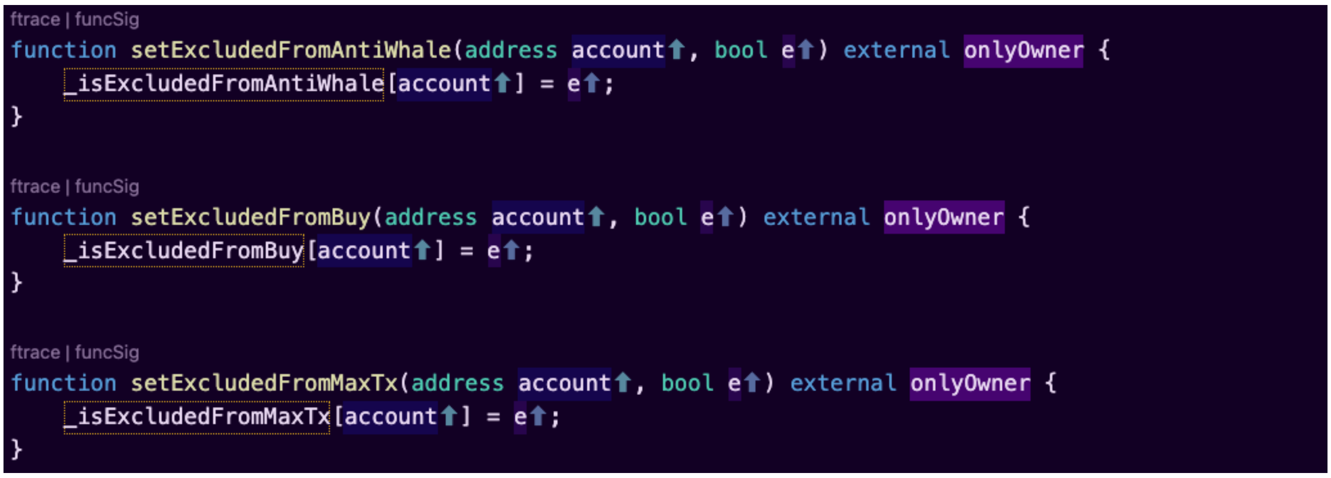
```
ftrace | funcSig
function setAntiWhaleThreshold(uint256 antiWhaleThreshold) external onlyOwner {
    _antiWhaleThreshold = antiWhaleThreshold;
}
```

- Owner can change _isAntiWhaleEnabled.



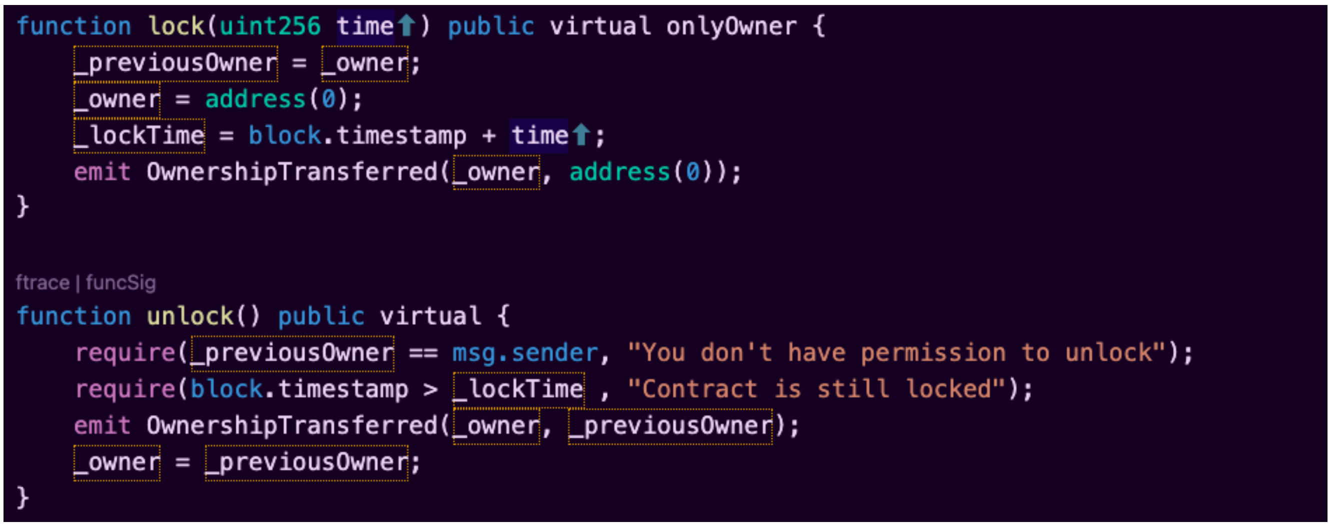
```
ftrace | funcSig
function setAntiWhaleEnabled(bool e) external onlyOwner {
    _isAntiWhaleEnabled = e;
}
```

- Owner can exclude from antiwhale, buy and maxTx.



```
ftrace | funcSig
function setExcludedFromAntiWhale(address account, bool e) external onlyOwner {
    _isExcludedFromAntiWhale[account] = e;
}

ftrace | funcSig
function setExcludedFromBuy(address account, bool e) external onlyOwner {
    _isExcludedFromBuy[account] = e;
}

ftrace | funcSig
function setExcludedFromMaxTx(address account, bool e) external onlyOwner {
    _isExcludedFromMaxTx[account] = 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=2769&add=0&type=lpdefi&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.*