TechRate
Blockchain solutions and consulting

# Smart Contract Security Audit

## Audit details:

| | |
|---|---|
| Audited project: | VELOREX |
| Deployer address: | 0x36266EBabECA70881e8473911a380dE270137615 |
| Client contacts: | VELOREX team |
| Blockchain: | Binance Smart Chain |
| Project website: | Not provided by the VELOREX team |

May, 2021
TechRate

# 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 VELOREX to perform an audit of smart contracts:

- *https://bscscan.com/address/0x7996cf3fbc0655131bd5a6d66e1ae2720f1de45e#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.

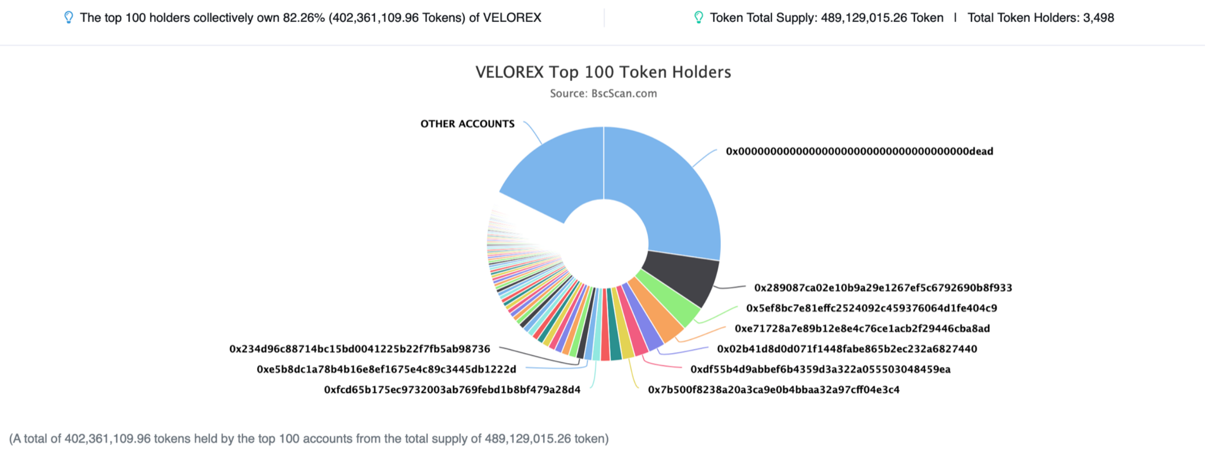
The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

# Contracts details

## Token contract details for 24.05.2021.

| | |
|---|---|
| Contract name: | VELOREX |
| Contract address: | 0x7996cF3FBc0655131BD5a6D66e1ae2720f1de45E |
| Total supply: | 489129015255055350 |
| Token ticker: | VEX |
| Decimals: | 9 |
| Token holders: | 3,498 |
| Transactions count: | 9,003 |
| Top 100 holders dominance: | 82.26% |
| Liquidity fee: | 100 |
| Tax fee: | 400 |
| Total fees: | 7378486407387033 |
| Uniswap V2 pair: | 0x289087ca02e10b9a29e1267ef5c6792690b8f933 |
| Contract deployer address: | 0x36266EBabECA70881e8473911a380dE270137615 |
| Contract’s current owner address: | 0x36266ebabeca70881e8473911a380de270137615 |

# VELOREX token distribution



The top 100 holders collectively own 82.26% (402,361,109.96 Tokens) of VELOREX

Token Total Supply: 489,129,015.26 Token | Total Token Holders: 3,498

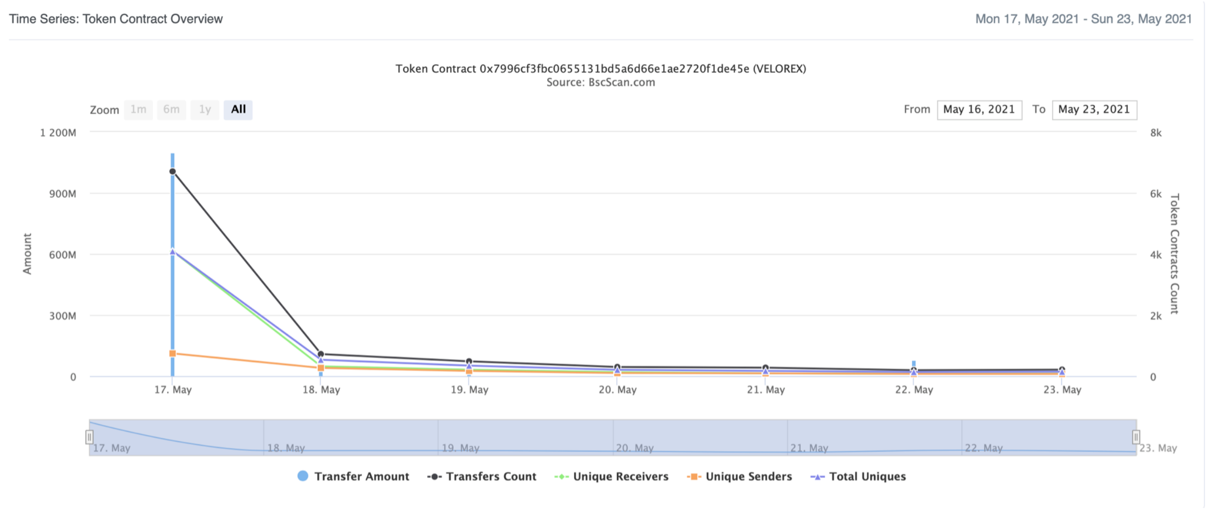
VELOREX Top 100 Token Holders

Source: BscScan.com

OTHER ACCOUNTS
0x000000000000000000000000000000000000dead
0x289087ca02e10b9a29e1267ef5c6792690b8f933
0x5ef8bc7e81effc2524092c459376064d1fe404c9
0xe71728a7e89b12e8e4c76ce1acb2f29446cba8ad
0x02b41d8d0d071f1448fabe865b2ec232a6827440
0xdf55b4d9abbef6b4359d3a322a055503048459ea
0x7b500f8238a20a3ca9e0b4bbaa32a97cff04e3c4
0x234d96c88714bc15bd0041225b22f7fb5ab98736
0xe5b8dc1a78b4b16e8ef1675e4c89c3445db1222d
0xfcd65b175ec9732003ab769febd1b8bf479a28d4

(A total of 402,361,109.96 tokens held by the top 100 accounts from the total supply of 489,129,015.26 token)

# VELOREX contract interaction details



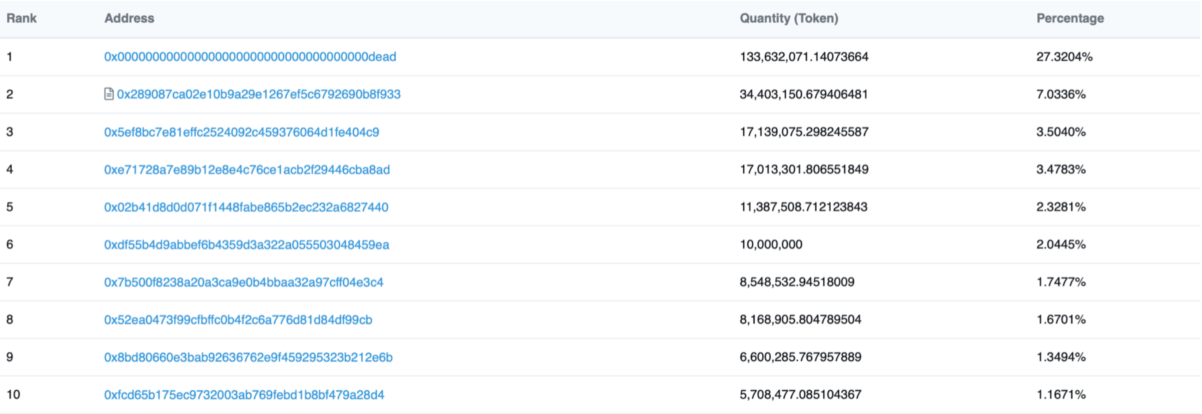
Time Series: Token Contract Overview

Mon 17, May 2021 - Sun 23, May 2021

Token Contract 0x7996cf3fbc0655131bd5a6d66e1ae2720f1de45e (VELOREX)

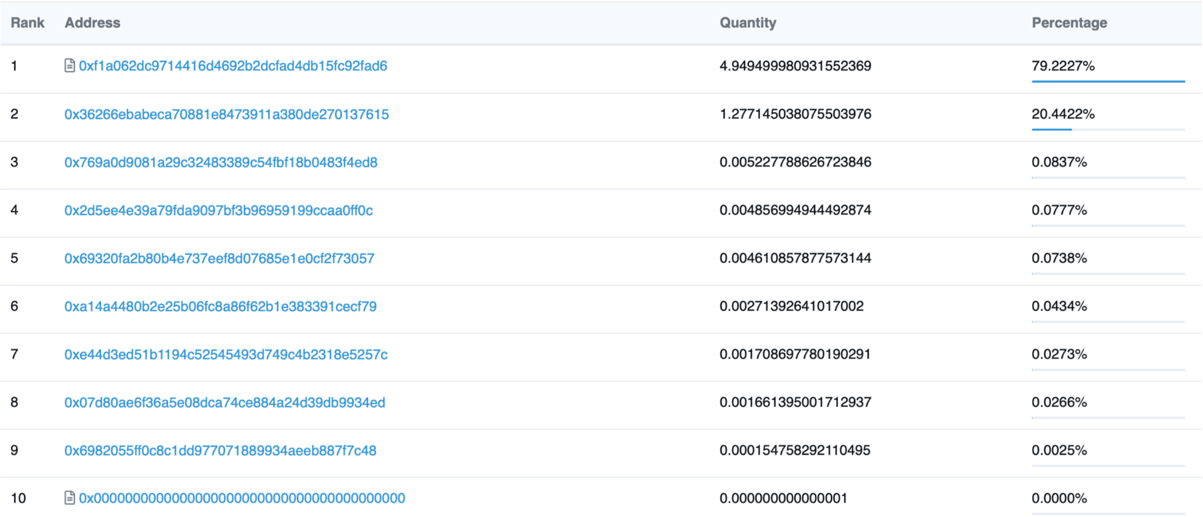
Source: BscScan.com

# VELOREX top 10 token holders



| Rank | Address | Quantity (Token) | Percentage |
|---|---|---|---|
| 1 | 0x000000000000000000000000000000000000dead | 133,632,071.14073664 | 27.3204% |
| 2 | 📄 0x289087ca02e10b9a29e1267ef5c6792690b8f933 | 34,403,150.679406481 | 7.0336% |
| 3 | 0x5ef8bc7e81effc2524092c459376064d1fe404c9 | 17,139,075.298245587 | 3.5040% |
| 4 | 0xe71728a7e89b12e8e4c76ce1acb2f29446cba8ad | 17,013,301.806551849 | 3.4783% |
| 5 | 0x02b41d8d0d071f1448fabe865b2ec232a6827440 | 11,387,508.712123843 | 2.3281% |
| 6 | 0xdf55b4d9abbef6b4359d3a322a055503048459ea | 10,000,000 | 2.0445% |
| 7 | 0x7b500f8238a20a3ca9e0b4bbaa32a97cff04e3c4 | 8,548,532.94518009 | 1.7477% |
| 8 | 0x52ea0473f99cfbffc0b4f2c6a776d81d84df99cb | 8,168,905.804789504 | 1.6701% |
| 9 | 0x8bd80660e3bab92636762e9f459295323b212e6b | 6,600,285.767957889 | 1.3494% |
| 10 | 0xfcd65b175ec9732003ab769febd1b8bf479a28d4 | 5,708,477.085104367 | 1.1671% |

# VELOREX LP token holders



| Rank | Address | Quantity | Percentage |
|---|---|---|---|
| 1 | 📄 0xf1a062dc9714416d4692b2dcfad4db15fc92fad6 | 4.949499980931552369 | 79.2227% |
| 2 | 0x36266ebabeca70881e8473911a380de270137615 | 1.277145038075503976 | 20.4422% |
| 3 | 0x769a0d9081a29c32483389c54fbf18b0483f4ed8 | 0.005227788626723846 | 0.0837% |
| 4 | 0x2d5ee4e39a79fda9097bf3b96959199ccaa0ff0c | 0.004856994944492874 | 0.0777% |
| 5 | 0x69320fa2b80b4e737eef8d07685e1e0cf2f73057 | 0.004610857877573144 | 0.0738% |
| 6 | 0xa14a4480b2e25b06fc8a86f62b1e383391cecf79 | 0.00271392641017002 | 0.0434% |
| 7 | 0xe44d3ed51b1194c52545493d749c4b2318e5257c | 0.001708697780190291 | 0.0273% |
| 8 | 0x07d80ae6f36a5e08dca74ce884a24d39db9934ed | 0.001661395001712937 | 0.0266% |
| 9 | 0x6982055ff0c8c1dd977071889934aeeb887f7c48 | 0.000154758292110495 | 0.0025% |
| 10 | 📄 0x0000000000000000000000000000000000000000 | 0.000000000000001 | 0.0000% |

# 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 #

+ VELOREX (Context, IBEP20, Ownable)
  - [Pub] <Constructor> #
    - modifiers: Ownable
  - [Ext] <Fallback> ($)
  - [Pub] deliver #
  - [Pub] reflectionFromToken
  - [Pub] tokenFromReflection
  - [Pub] excludeFromReward #
    - modifiers: onlyOwner
  - [Ext] includeInReward #
    - modifiers: onlyOwner
  - [Pub] totalFees
  - [Pub] totalBurn
  - [Pub] excludeFromFee #
    - modifiers: onlyOwner
  - [Pub] includeInFee #
    - modifiers: onlyOwner
  - [Ext] setTaxFeePercent #
    - modifiers: onlyOwner
  - [Ext] setLiquidityFeePercent #
    - modifiers: onlyOwner
  - [Ext] setBurnFeePercent #
    - modifiers: onlyOwner
  - [Ext] setMaxTxPercent #
    - modifiers: onlyOwner
  - [Ext] setMinLiquidityPercent #
    - modifiers: onlyOwner
  - [Pub] setSwapAndLiquifyEnabled #
    - modifiers: onlyOwner
  - [Pub] isExcludedFromFee
  - [Pub] isExcludedFromReward
  - [Ext] setIsExcludedFromSwapAndLiquify #
    - modifiers: onlyOwner
  - [Ext] setUniswapRouter #
    - modifiers: onlyOwner
  - [Ext] setUniswapPair #
    - modifiers: onlyOwner

- [Pub] name
- [Pub] symbol
- [Pub] decimals
- [Pub] totalSupply
- [Pub] balanceOf
- [Pub] transfer #
- [Pub] allowance
- [Pub] approve #
- [Prv] _approve #
- [Pub] transferFrom #
- [Pub] increaseAllowance #
- [Pub] decreaseAllowance #
- [Prv] _transfer #
- [Prv] swapAndLiquify #
  - modifiers: lockTheSwap
- [Prv] swapTokensForBnb #
- [Prv] _tokenTransfer #
- [Prv] _transferStandard #
- [Prv] _transferBothExcluded #
- [Prv] _transferToExcluded #
- [Prv] _transferFromExcluded #
- [Prv] _reflectFee #
- [Prv] _getTValues
- [Prv] _getRValues
- [Prv] _getRate
- [Prv] _getCurrentSupply
- [Prv] takeTransactionFee #

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

# Issues Checking Status

| № | Issue description. | Checking status |
|---|---|---|
| 1 | Compiler errors. | Passed |
| 2 | Race conditions and Reentrancy.<br>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

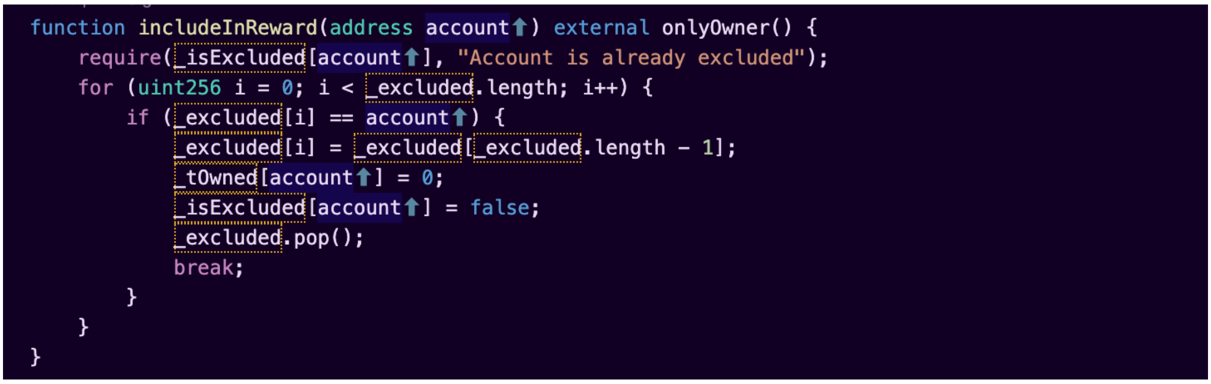
No medium severity issues found.

## Low Severity Issues

### 1. Out of gas

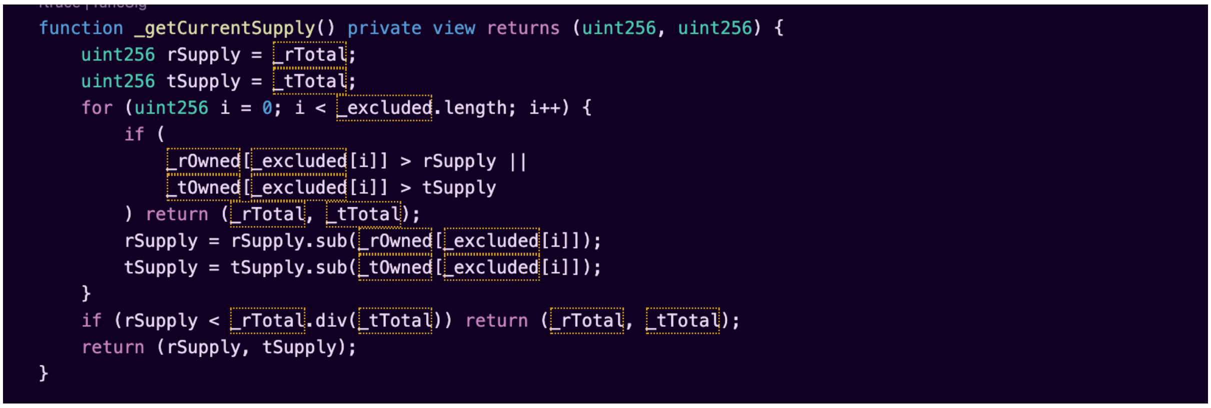
Issue:

- 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.



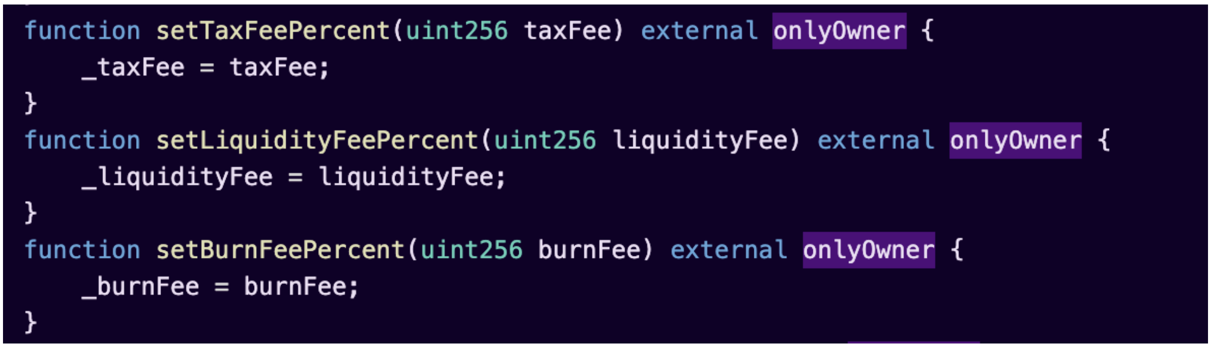
```
function _getCurrentSupply() private view returns (uint256, uint256) {
    uint256 rSupply = _rTotal;
    uint256 tSupply = _tTotal;
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (
            _rOwned[_excluded[i]] > rSupply ||
            _tOwned[_excluded[i]] > tSupply
      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

Use EnumerableSet instead of array or do not use long arrays.

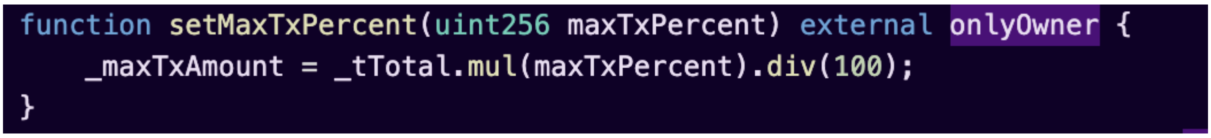
## Owner privileges (In the period when the owner is not renounced)

❑ Owner can change the tax, burn and liquidity fee.



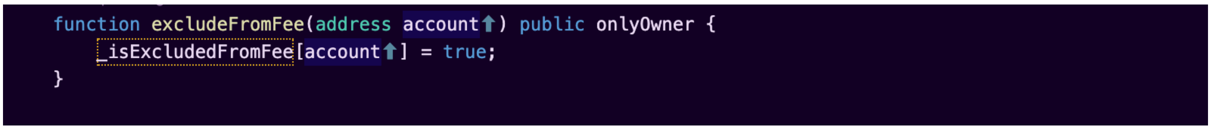
```
function setTaxFeePercent(uint256 taxFee) external onlyOwner {
    _taxFee = taxFee;
}
function setLiquidityFeePercent(uint256 liquidityFee) external onlyOwner {
    _liquidityFee = liquidityFee;
}
function setBurnFeePercent(uint256 burnFee) external onlyOwner {
    _burnFee = burnFee;
}
```

❑ Owner can change the maximum transaction amount.



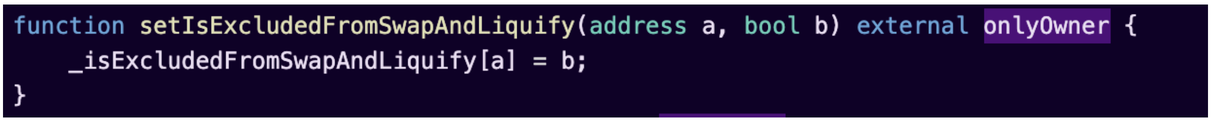
```
function setMaxTxPercent(uint256 maxTxPercent) external onlyOwner {
    _maxTxAmount = _tTotal.mul(maxTxPercent).div(100);
}
```

❑ Owner can exclude from the fee.



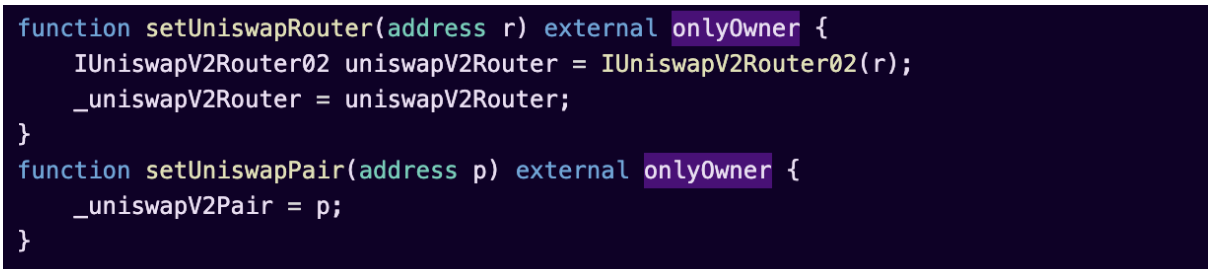
```
function excludeFromFee(address account⬆) public onlyOwner {
    _isExcludedFromFee[account⬆] = true;
}
```

❑ Owner can include & exclude from swapAndLiquify (swapAndLiquify won't be called)



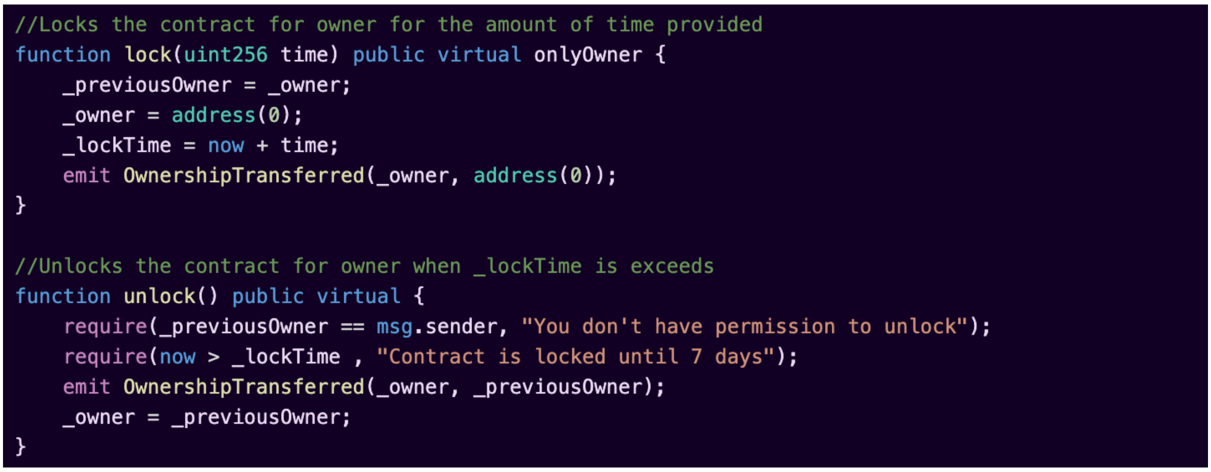
```
function setIsExcludedFromSwapAndLiquify(address a, bool b) external onlyOwner {
    _isExcludedFromSwapAndLiquify[a] = b;
}
```

❑ Owner can change uniswap router & pair



```
function setUniswapRouter(address r) external onlyOwner {
    IUniswapV2Router02 uniswapV2Router = IUniswapV2Router02(r);
    _uniswapV2Router = uniswapV2Router;
}
function setUniswapPair(address p) external onlyOwner {
    _uniswapV2Pair = p;
}
```

❑ Owner can lock and unlock. By the way, using these functions the owner could leave as owner even after the ownership was renounced.

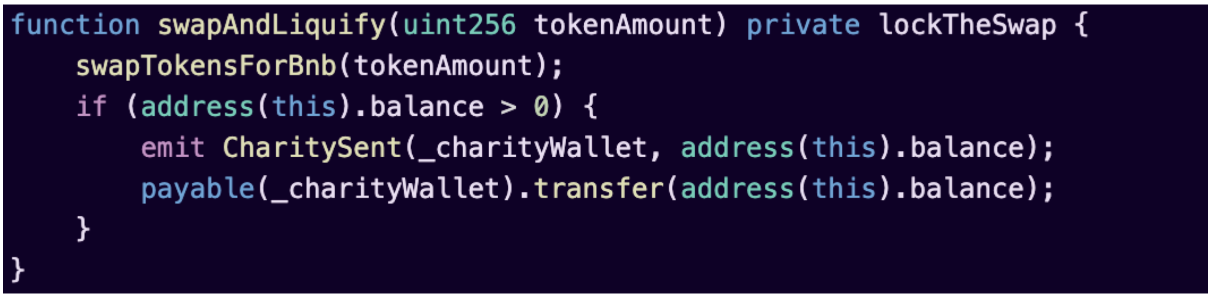


```
//Locks the contract for owner for the amount of time provided
function lock(uint256 time) public virtual onlyOwner {
    _previousOwner = _owner;
    _owner = address(0);
    _lockTime = now + time;
    emit OwnershipTransferred(_owner, address(0));
}

//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;
}
```

## Notes

- ❏ swapAndLiquify() function sends all swap balance to charity wallet.



```
function swapAndLiquify(uint256 tokenAmount) private lockTheSwap {
    swapTokensForBnb(tokenAmount);
    if (address(this).balance > 0) {
        emit CharitySent(_charityWallet, address(this).balance);
        payable(_charityWallet).transfer(address(this).balance);
    }
}
```

# Conclusion

**Smart contracts contain low severity issues. LP pair contract is not checked.**

**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.*