Techrate1 Techrate

TechRate

AUDIT COMPANY

Smart Contract Security Audit

TechRate

July, 2021

Audit Details

Audited project

NinjaDoge

Deployer address

0xc545dC77C7A88B3c379737572C25E4cA4651c1f4

Client contacts:

NinjaDoge team

Blockchain

Binance Smart Chain

Project website:

https://www.ninjadoge.org/

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 NinjaDoge to perform an audit of smart contracts:
https://bscscan.com/address/0xe218DcF32F9bb64648D64Ef2AE85cF1C63C5aC74#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.

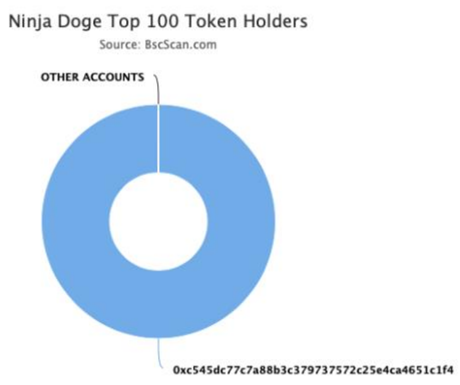
Contracts Details

Token contract details for 23.07.2021

Contract name	NinjaDoge
Contract address	0xe218DcF32F9bb64648D64Ef2AE85cF1C63C5aC74
Total supply	1,000,000,000,000,000
Token ticker	$NINJADOGE
Decimals	9
Token holders	1
Transactions count	1
Top 100 holders dominance	100.00%
Liquidity fee	5
Reward fee	5
Total tax fees	17
Uniswap V2 pair	0xc889263c7f9ed0d67007ba8b90c327a01bcffcc9
Contract deployer address	0xc545dC77C7A88B3c379737572C25E4cA4651c1f4
Contract's current owner address	0x0000000000000000000000000000000000000000

NinjaDoge Token Distribution

The top 100 holders collectively own 100.00% (1,000,000,000,000,000.00 Tokens) of Ninja Doge | Token Total Supply: 1,000,000,000,000,000.00 Token | Total Token Holders: 1



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

NinjaDoge Top 10 Token Holders

Rank	Address	Quantity (Token)	Percent
1.	0xc545dc77c7a88b3c379737572c25e4ca4651c1f4	1,000,000,000,000,000	100.0000%

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)
  - [Pub] <Constructor> #
  - [Pub] owner
  - [Pub] renounceOwnership #
    - modifiers: onlyOwner
  - [Pub] transferOwnership #
    - modifiers: onlyOwner
  - [Pub] getUnlockTime
  - [Pub] getTime
  - [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] 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 #
```

```
+  NinjaDoge (Context, IERC20, Ownable)
  - [Pub] <Constructor> #
  - [Pub] name
  - [Pub] symbol
  - [Pub] decimals
  - [Pub] totalSupply
  - [Pub] balanceOf
  - [Pub] allowance
  - [Pub] increaseAllowance #
  - [Pub] decreaseAllowance #
  - [Pub] minimumTokensBeforeSwapAmount
  - [Pub] totalTax
  - [Pub] approve #
  - [Prv] _approve #
  - [Pub] isExcludedFromFee
  - [Pub] excludeFromFee #
    - modifiers: onlyOwner
  - [Pub] includeInFee #
    - modifiers: onlyOwner
  - [Ext] setTaxes #
    - modifiers: onlyOwner
  - [Ext] setMaxTxAmount #
    - modifiers: onlyOwner
  - [Ext] setNumTokensBeforeSwap #
    - modifiers: onlyOwner
  - [Ext] setMarketingWalletAddress #
    - modifiers: onlyOwner
  - [Ext] setRewardWalletAddress #
    - modifiers: onlyOwner
  - [Pub] setSwapAndLiquifyEnabled #
    - modifiers: onlyOwner
  - [Pub] setSwapAndLiquifyByLimitOnly #
    - modifiers: onlyOwner
  - [Pub] getCirculatingSupply
  - [Ext] prepareForPreSale #
    - modifiers: onlyOwner
  - [Ext] prepareForLaunch #
    - modifiers: onlyOwner
  - [Prv] transferToAddressETH #
  - [Pub] changeRouterVersion #
    - modifiers: onlyOwner
  - [Ext] <Fallback> ($)
  - [Pub] transfer #
  - [Pub] transferFrom #
  - [Prv] _transfer #
  - [Int] _basicTransfer #
  - [Prv] swapAndLiquify #
    - modifiers: lockTheSwap
  - [Prv] swapTokensForEth #
  - [Prv] addLiquidity #
  - [Int] takeFee #

($) = 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.	Passed
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

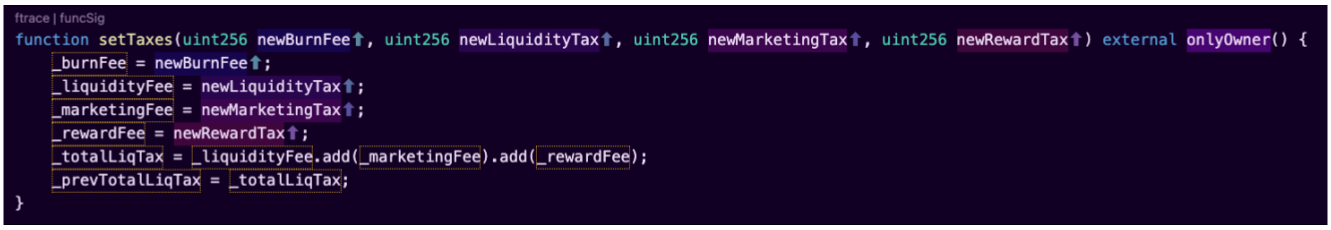
No medium severity issues found.

Low Severity Issues

No low severity issues found.

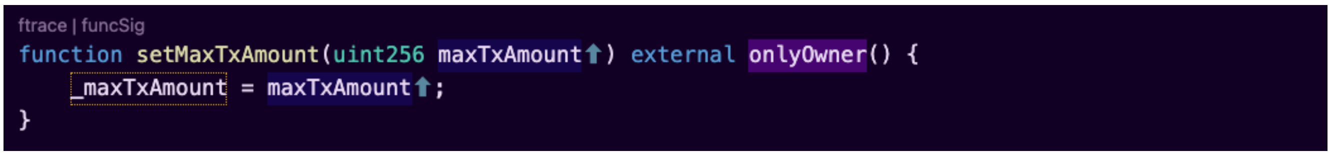
Owner privileges (In the period when the owner is not renounced)

- Owner can change the burn, marketing, reward and liquidity fee.



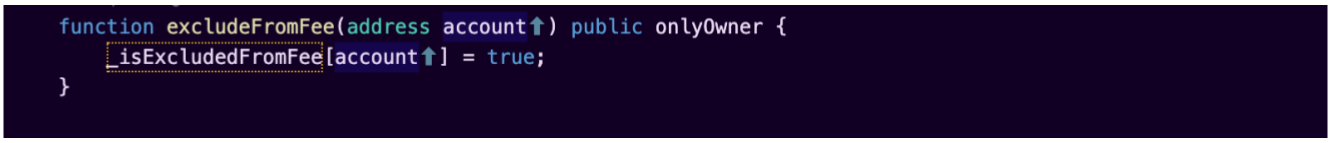
```
ftrace | funcSig
function setTaxes(uint256 newBurnFee, uint256 newLiquidityTax, uint256 newMarketingTax, uint256 newRewardTax) external onlyOwner() {
    _burnFee = newBurnFee;
    _liquidityFee = newLiquidityTax;
    _marketingFee = newMarketingTax;
    _rewardFee = newRewardTax;
    _totalLiqTax = _liquidityFee.add(_marketingFee).add(_rewardFee);
    _prevTotalLiqTax = _totalLiqTax;
}
```

- Owner can change the maximum transaction amount.



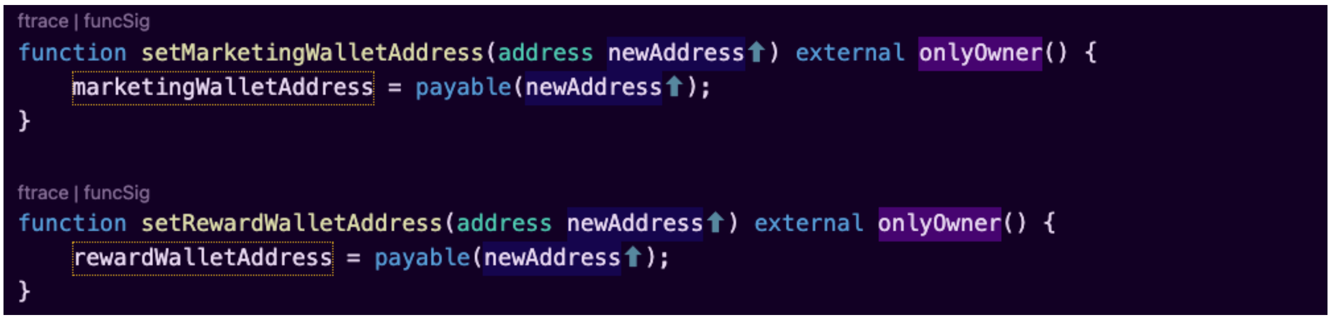
```
ftrace | funcSig
function setMaxTxAmount(uint256 maxTxAmount) external onlyOwner() {
    _maxTxAmount = maxTxAmount;
}
```

- Owner can exclude from the fee.



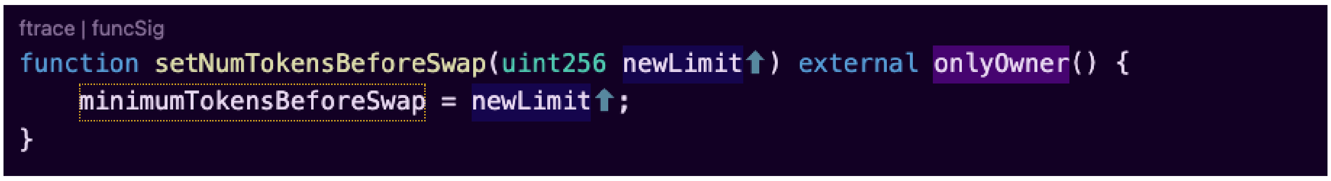
```
    function excludeFromFee(address account) public onlyOwner {
        _isExcludedFromFee[account] = true;
    }
```

- **Owner can marketing and reward wallets.**



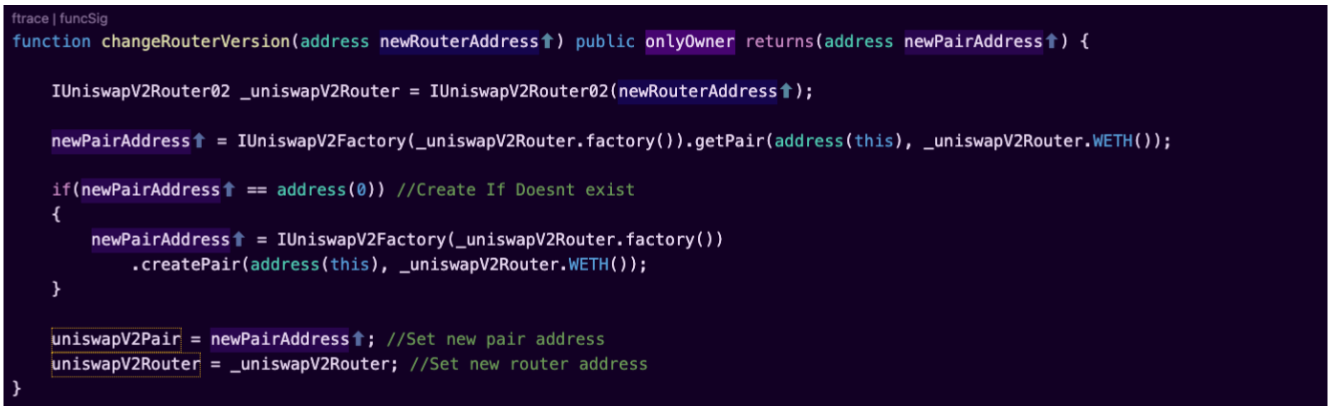
```
ftrace | funcSig
function setMarketingWalletAddress(address newAddress) external onlyOwner() {
    marketingWalletAddress = payable(newAddress);
}

ftrace | funcSig
function setRewardWalletAddress(address newAddress) external onlyOwner() {
    rewardWalletAddress = payable(newAddress);
}
```

- **Owner can change minimum number of tokens before swap.**



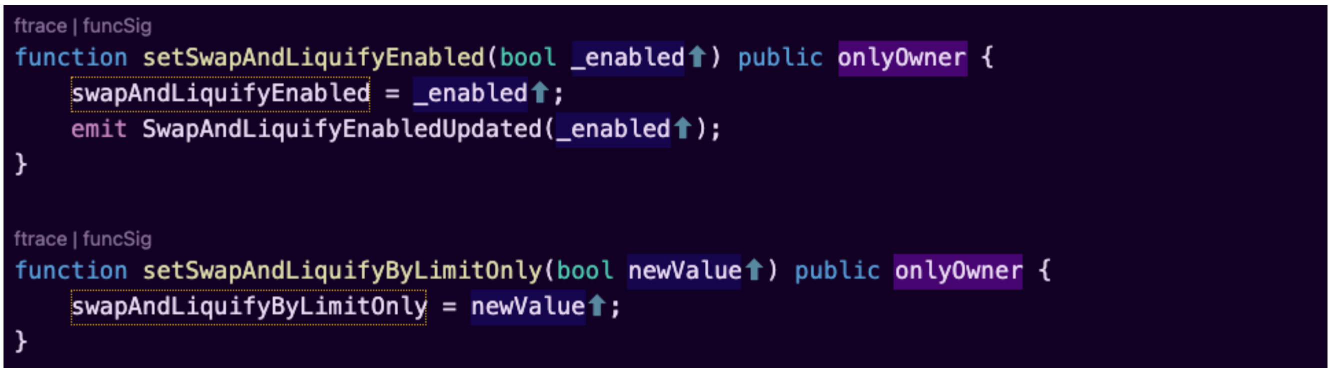
```
ftrace | funcSig
function setNumTokensBeforeSwap(uint256 newLimit) external onlyOwner() {
    minimumTokensBeforeSwap = newLimit;
}
```

- **Owner can change Uniswap router address.**



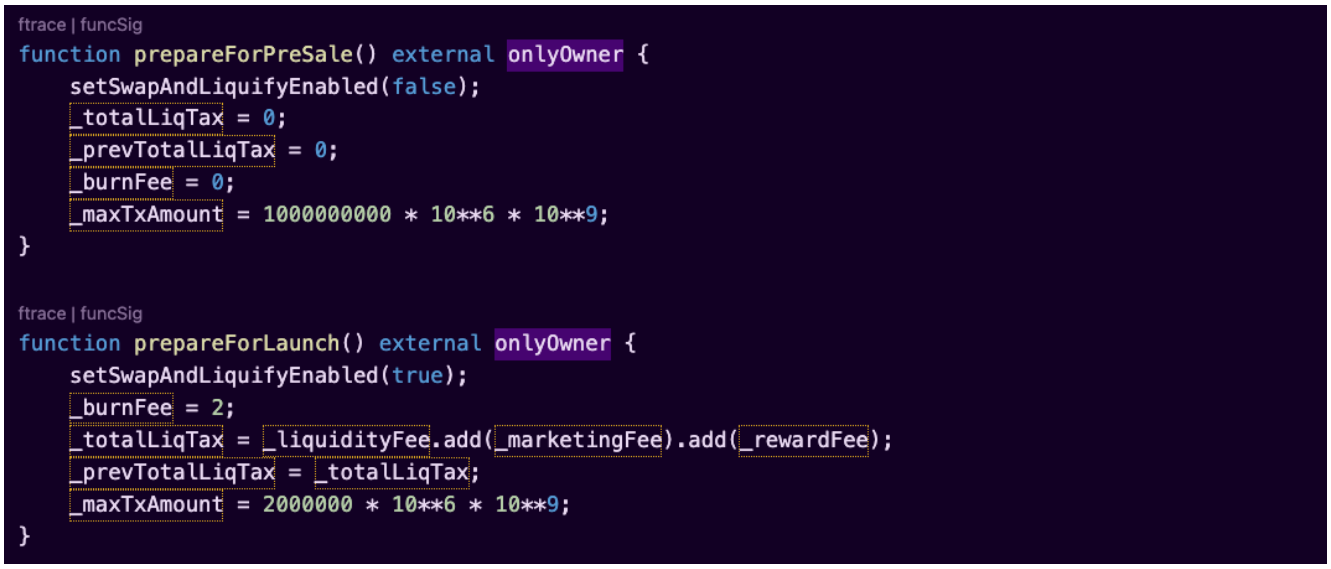
```
ftrace | funcSig
function changeRouterVersion(address newRouterAddress) public onlyOwner returns(address newPairAddress) {

    IUniswapV2Router02 _uniswapV2Router = IUniswapV2Router02(newRouterAddress);

    newPairAddress = IUniswapV2Factory(_uniswapV2Router.factory()).getPair(address(this), _uniswapV2Router.WETH());

    if(newPairAddress == address(0)) //Create If Doesnt exist
    {
        newPairAddress = IUniswapV2Factory(_uniswapV2Router.factory())
            .createPair(address(this), _uniswapV2Router.WETH());
    }

    uniswapV2Pair = newPairAddress; //Set new pair address
    uniswapV2Router = _uniswapV2Router; //Set new router address
}
```

- **Owner can change swap and liquify settings.**



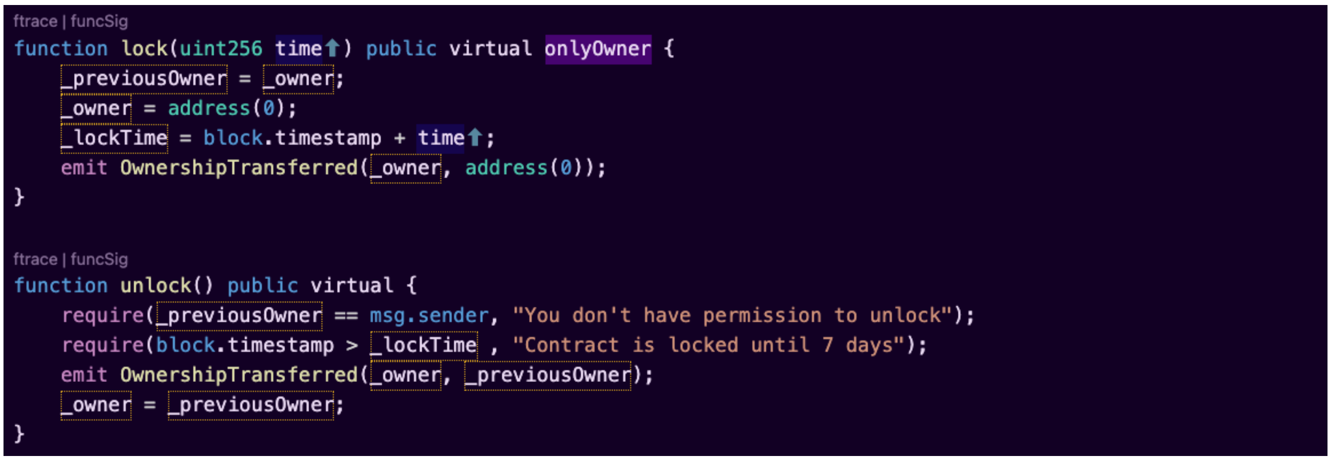
```
ftrace | funcSig
function setSwapAndLiquifyEnabled(bool _enabled) public onlyOwner {
    swapAndLiquifyEnabled = _enabled;
    emit SwapAndLiquifyEnabledUpdated(_enabled);
}

ftrace | funcSig
function setSwapAndLiquifyByLimitOnly(bool newValue) public onlyOwner {
    swapAndLiquifyByLimitOnly = newValue;
}
```

- **Owner can enable presale and launch presets.**



```
ftrace | funcSig
function prepareForPreSale() external onlyOwner {
    setSwapAndLiquifyEnabled(false);
    _totalLiqTax = 0;
    _prevTotalLiqTax = 0;
    _burnFee = 0;
    _maxTxAmount = 1000000000 * 10**6 * 10**9;
}

ftrace | funcSig
function prepareForLaunch() external onlyOwner {
    setSwapAndLiquifyEnabled(true);
    _burnFee = 2;
    _totalLiqTax = _liquidityFee.add(_marketingFee).add(_rewardFee);
    _prevTotalLiqTax = _totalLiqTax;
    _maxTxAmount = 2000000 * 10**6 * 10**9;
}
```

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



```
ftrace | funcSig
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 locked until 7 days");
    emit OwnershipTransferred(_owner, _previousOwner);
    _owner = _previousOwner;
}
```

Conclusion

Smart contracts do not contain high 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.