Techrate1 Techrate

TechRate

AUDIT COMPANY

Smart Contract Security Audit

TechRate

October, 2021

Audit Details

Audited project

Metastake

Deployer address

0xEd7F4873a8bE16d166f5CeE1Cc8FEd9Df087C4F3

Client contacts:

Metastake team

Blockchain

Binance Smart Chain

Project website:

https://metastake.net/

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 Metastake to perform an audit of smart contracts:
https://bscscan.com/address/0x8ffe73b5730fead733f596cbe0f2681bda66571f#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.

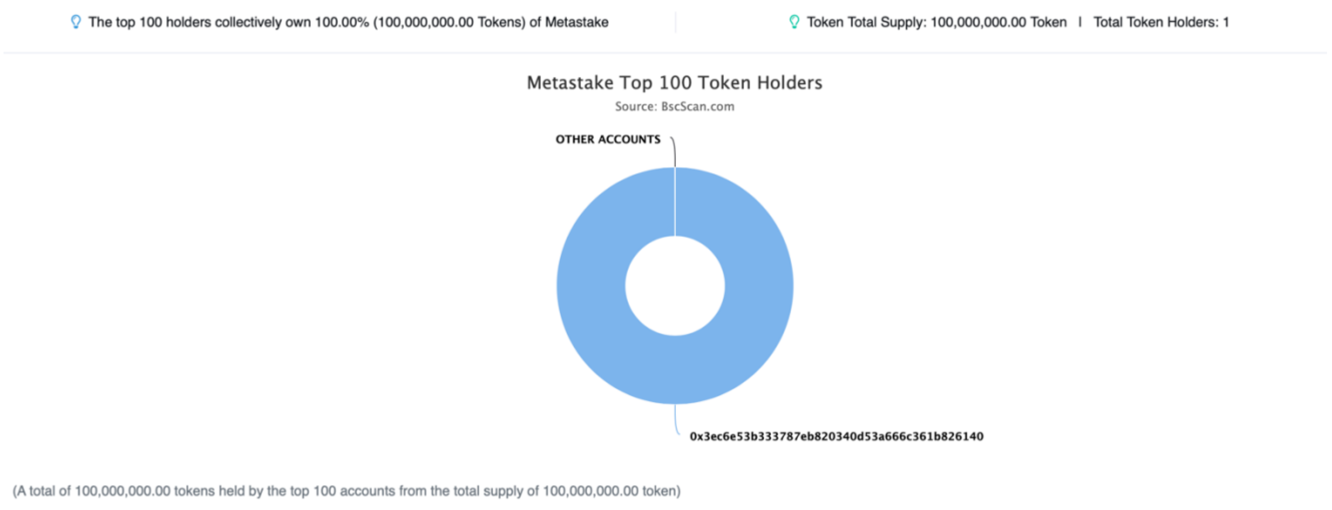
The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

Contracts Details

Token contract details for 09.10.2021

Contract name	Metastake
Contract address	0x8fFE73b5730FEAd733f596cBe0f2681bdA66571F
Total supply	100,000,000
Token ticker	MST
Decimals	18
Token holders	1
Transactions count	2
Top 100 holders dominance	100.00%
Dividend token	0xe9e7cea3dedca5984780bafc599bd69add087d56
Contract deployer address	0xEd7F4873a8bE16d166f5CeE1Cc8FEd9Df087C4F3
Contract’s current owner address	0x3ec6e53b333787eb820340d53a666c361b826140

Metastake Token Distribution


The top 100 holders collectively own 100.00% (100,000,000.00 Tokens) of Metastake

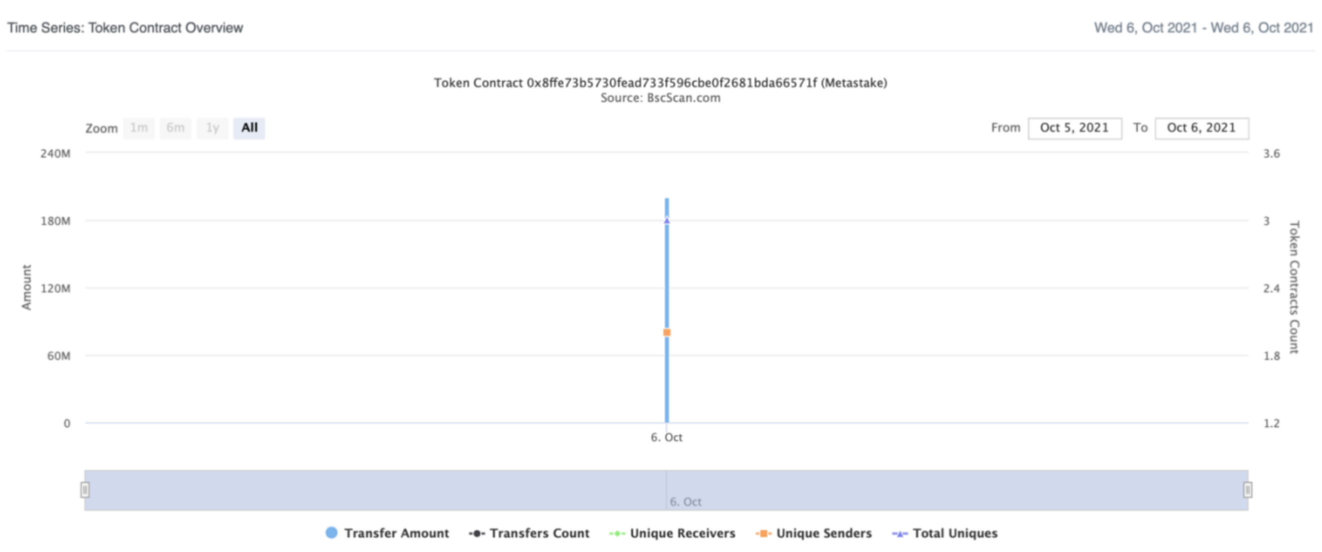
Token Total Supply: 100,000,000.00 Token | Total Token Holders: 1

Metastake Top 100 Token Holders

Source: BscScan.com

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

Metastake Contract Interaction Details

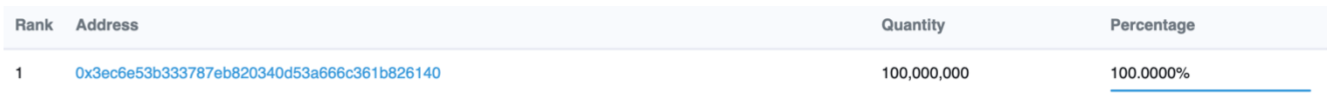

Time Series: Token Contract Overview

Wed 6, Oct 2021 - Wed 6, Oct 2021

Token Contract 0x8ffe73b5730fead733f596cbe0f2681bda66571f (Metastake)

Source: BscScan.com

Metastake Top 10 Token Holders



Rank	Address	Quantity	Percentage
1	0x3ec6e53b333787eb820340d53a666c361b826140	100,000,000	100.0000%

Contract functions details

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

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

+ [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 #

+ [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 #

+ [Lib] IterableMapping
  - [Pub] get
  - [Pub] getIndexOfKey
  - [Pub] getKeyAtIndex
  - [Pub] size
  - [Pub] set #
  - [Pub] remove #

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

+ [Int] IDividendPayingTokenOptional
  - [Ext] withdrawableDividendOf
  - [Ext] withdrawnDividendOf
  - [Ext] accumulativeDividendOf

+ [Int] IDividendPayingToken
  - [Ext] dividendOf
  - [Ext] distributeDividends ($)
  - [Ext] withdrawDividend #

+ [Lib] SafeMathInt

- [Int] mul
- [Int] div
- [Int] sub
- [Int] add
- [Int] toUint256Safe

+ [Lib] SafeMathUint
  - [Int] toInt256Safe

+ ERC20 (Context, IERC20)
  - [Pub] <Constructor> #
  - [Pub] name
  - [Pub] symbol
  - [Pub] decimals
  - [Pub] totalSupply
  - [Pub] balanceOf
  - [Pub] transfer #
  - [Pub] allowance
  - [Pub] approve #
  - [Pub] transferFrom #
  - [Pub] increaseAllowance #
  - [Pub] decreaseAllowance #
  - [Int] _transfer #
  - [Int] _mint #
  - [Int] _burn #
  - [Int] _approve #
  - [Int] _setupDecimals #
  - [Int] _beforeTokenTransfer #

+ [Lib] SafeMath
  - [Int] tryAdd
  - [Int] trySub
  - [Int] tryMul
  - [Int] tryDiv
  - [Int] tryMod
  - [Int] add
  - [Int] sub
  - [Int] mul
  - [Int] div
  - [Int] mod
  - [Int] sub
  - [Int] div
  - [Int] mod

+ DividendPayingToken (ERC20, IDividendPayingToken, IDividendPayingTokenOptional)
  - [Pub] <Constructor> #
    - modifiers: ERC20
  - [Ext] <Fallback> ($)
  - [Pub] updateMasterContract #
    - modifiers: onlyMaster
  - [Pub] distributeDividends ($)
  - [Pub] distributeTokenDividends #
    - modifiers: onlyMaster
  - [Pub] withdrawDividend #

- [Int] _withdrawDividendOfUser #
- [Pub] dividendOf
- [Pub] withdrawableDividendOf
- [Pub] withdrawnDividendOf
- [Pub] accumulativeDividendOf
- [Int] _transfer #
- [Int] _mint #
- [Int] _burn #
- [Int] _setBalance #

+ MST (ERC20, Ownable)
  - [Pub] <Constructor> #
    - modifiers: ERC20
  - [Ext] <Fallback> ($)
  - [Ext] swapAndLiquifyOwner #
    - modifiers: onlyOwner
  - [Int] restoreFees #
  - [Ext] updatedividendTime #
    - modifiers: onlyOwner
  - [Ext] updateBuyBackMode #
    - modifiers: onlyOwner
  - [Ext] updateisTransferDisabled #
    - modifiers: onlyOwner
  - [Ext] updateTradingEnabledTime #
    - modifiers: onlyOwner
  - [Ext] updateMinimumBalanceForDividends #
    - modifiers: onlyOwner
  - [Ext] updateMaxWalletAmount #
    - modifiers: onlyOwner
  - [Ext] updateSwapAtAmount #
    - modifiers: onlyOwner
  - [Ext] updateTokenForDividend #
    - modifiers: onlyOwner
  - [Ext] updateMarketAddress #
    - modifiers: onlyOwner
  - [Ext] updateCharityAddress #
    - modifiers: onlyOwner
  - [Ext] updateBuyBackAddress #
    - modifiers: onlyOwner
  - [Ext] updateMarketTokenFeeAddress #
    - modifiers: onlyOwner
  - [Ext] updateCharityTokenFeeAddress #
    - modifiers: onlyOwner
  - [Ext] updateBuyBackTokenFeeAddress #
    - modifiers: onlyOwner
  - [Ext] updateFees #
    - modifiers: onlyOwner
  - [Ext] updateBuyFees #
    - modifiers: onlyOwner
  - [Ext] updateSellFees #
    - modifiers: onlyOwner
  - [Ext] whitelistDxSale #
    - modifiers: onlyOwner
  - [Ext] updateDividendTracker #
    - modifiers: onlyOwner

  - [Ext] updateUniswapV2Router #
    - modifiers: onlyOwner
  - [Pub] excludeFromFees #
    - modifiers: onlyOwner
  - [Pub] excludeFromDividends #
    - modifiers: onlyOwner
  - [Pub] enableDividends #
    - modifiers: onlyOwner
  - [Ext] excludeMultipleAccountsFromFees #
    - modifiers: onlyOwner
  - [Ext] setAutomatedMarketMakerPair #
    - modifiers: onlyOwner
  - [Prv] _setAutomatedMarketMakerPair #
  - [Pub] updateGasForProcessing #
    - modifiers: onlyOwner
  - [Ext] updateClaimWait #
    - modifiers: onlyOwner
  - [Ext] getClaimWait
  - [Ext] getTotalDividendsDistributed
  - [Pub] isExcludedFromFees
  - [Pub] withdrawableDividendOf
  - [Pub] dividendTokenBalanceOf
  - [Ext] getAccountDividendsInfo
  - [Ext] getAccountDividendsInfoAtIndex
  - [Ext] processDividendTracker #
  - [Ext] claim #
  - [Ext] getLastProcessedIndex
  - [Ext] getNumberOfDividendTokenHolders
  - [Pub] getTradingIsEnabled
  - [Prv] swapAndLiquify #
  - [Prv] swapEthForTokens #
  - [Prv] swapTokensForEth #
  - [Prv] swapTokensForTokens #
  - [Int] addLiquidity #
  - [Prv] swapAndSendDividends #
  - [Int] _transfer #

+ MSTDividendTracker (DividendPayingToken, Ownable)
  - [Pub] <Constructor> #
    - modifiers: DividendPayingToken
  - [Ext] updateMinimumBalanceForDividends #
    - modifiers: onlyOwner
  - [Ext] updateTokenForDividend #
    - modifiers: onlyOwner
  - [Int] _transfer #
  - [Pub] withdrawDividend #
  - [Ext] excludeFromDividends #
    - modifiers: onlyOwner
  - [Ext] enableDividends #
    - modifiers: onlyOwner
  - [Ext] updateClaimWait #
    - modifiers: onlyOwner
  - [Ext] getLastProcessedIndex
  - [Ext] getNumberOfTokenHolders
  - [Pub] getAccount

  - [Pub] getAccountAtIndex
  - [Prv] canAutoClaim
  - [Ext] setBalance #
    - modifiers: onlyOwner
  - [Pub] process #
  - [Pub] processAccount #
    - 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

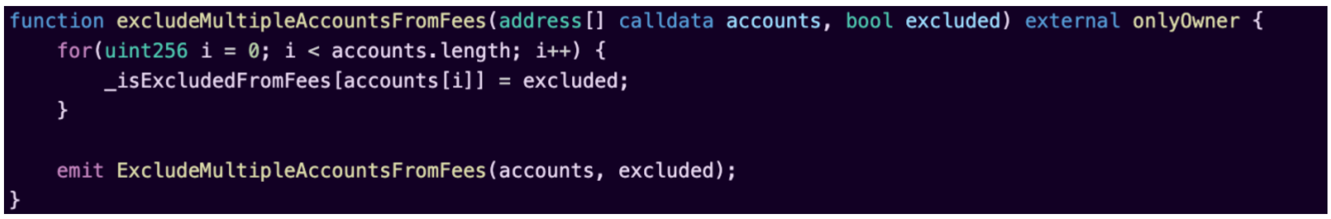
No medium severity issues found.

Low Severity Issues

1. Out of gas

Issue:

- The function excludeMultipleAccountsFromFees() uses the loop to exclude multiple accounts from fees. Function will be aborted with OUT_OF_GAS exception if there will be a long addresses list.



```
function excludeMultipleAccountsFromFees(address[] calldata accounts, bool excluded) external onlyOwner {
    for(uint256 i = 0; i < accounts.length; i++) {
        _isExcludedFromFees[accounts[i]] = excluded;
    }

    emit ExcludeMultipleAccountsFromFees(accounts, excluded);
}
```

Recommendation:

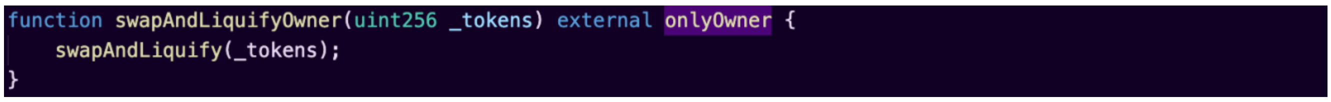
Be careful about accounts array length.

Notes:

- Owner can change dividend tracker that could be not audited and some functions may work in different ways.

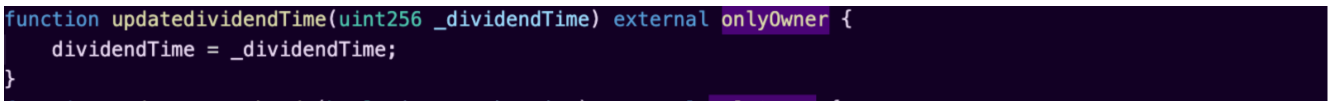
Owner privileges (In the period when the owner is not renounced)

- Owner can swap and liquify.



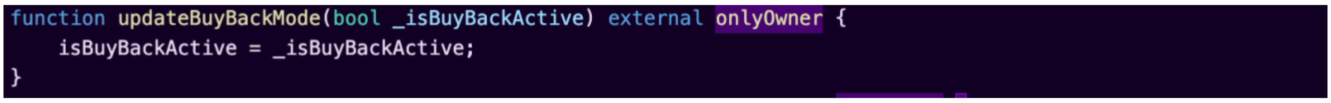
```
function swapAndLiquifyOwner(uint256 _tokens) external onlyOwner {
    swapAndLiquify(_tokens);
}
```

- Owner can update dividend time.



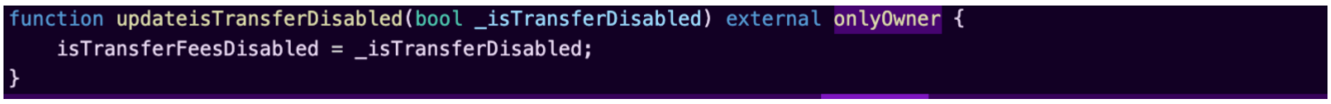
```
function updatedividendTime(uint256 _dividendTime) external onlyOwner {
    dividendTime = _dividendTime;
}
```

- Owner can enable / disable buy back.



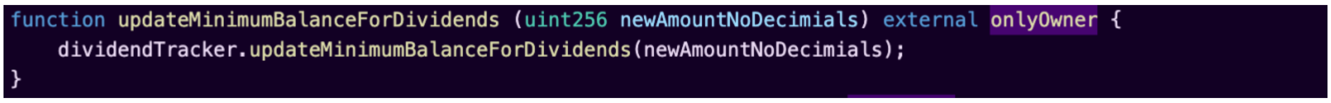
```
function updateBuyBackMode(bool _isBuyBackActive) external onlyOwner {
    isBuyBackActive = _isBuyBackActive;
}
```

- Owner can enable / disable transfer fee.



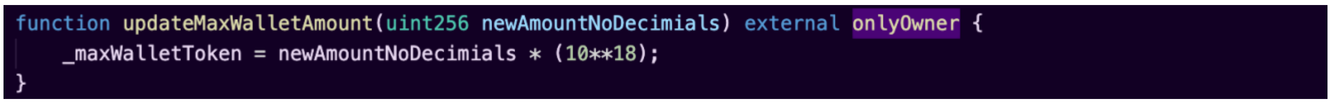
```
function updateisTransferDisabled(bool _isTransferDisabled) external onlyOwner {
    isTransferFeesDisabled = _isTransferDisabled;
}
```

- Owner can change minimum balance for dividends.



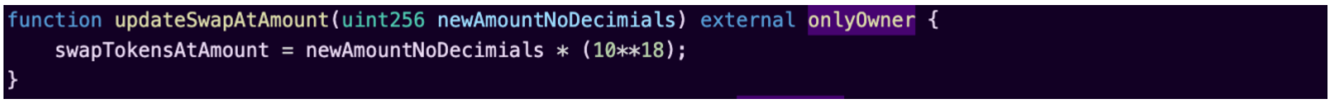
```
function updateMinimumBalanceForDividends (uint256 newAmountNoDecimials) external onlyOwner {
    dividendTracker.updateMinimumBalanceForDividends(newAmountNoDecimials);
}
```

- Owner can change maximum wallet token amount.



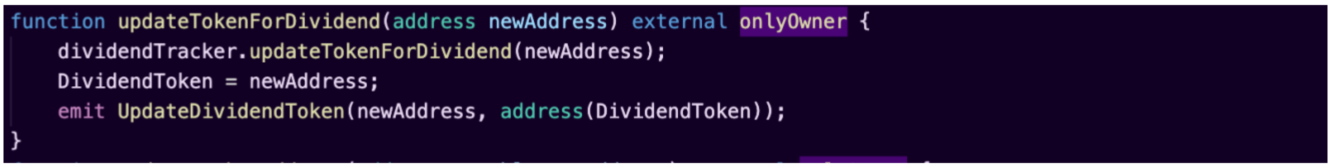
```
function updateMaxWalletAmount(uint256 newAmountNoDecimials) external onlyOwner {
    _maxWalletToken = newAmountNoDecimials * (10**18);
}
```

- Owner can change amount for swap token start.



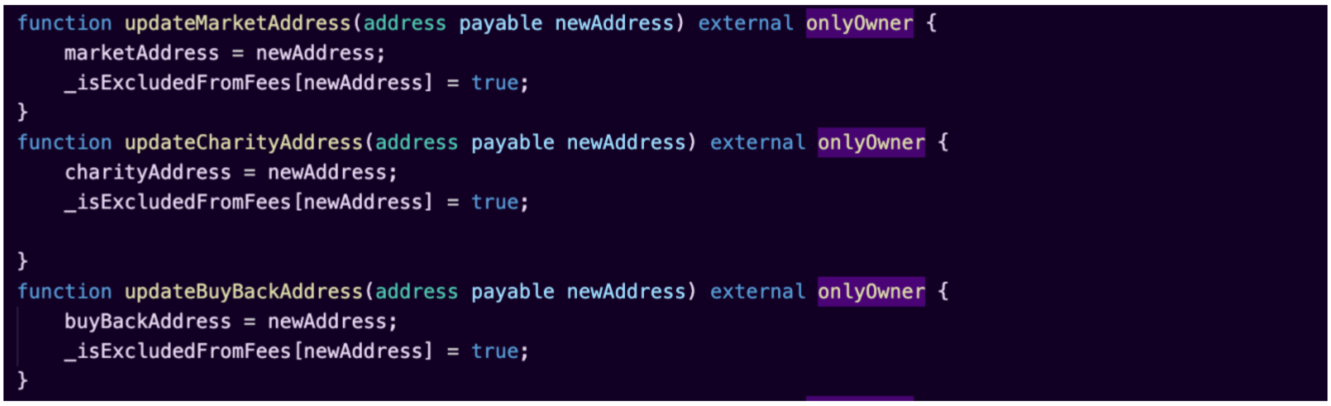
```
function updateSwapAtAmount(uint256 newAmountNoDecimials) external onlyOwner {
    swapTokensAtAmount = newAmountNoDecimials * (10**18);
}
```

- Owner can change dividend token address.



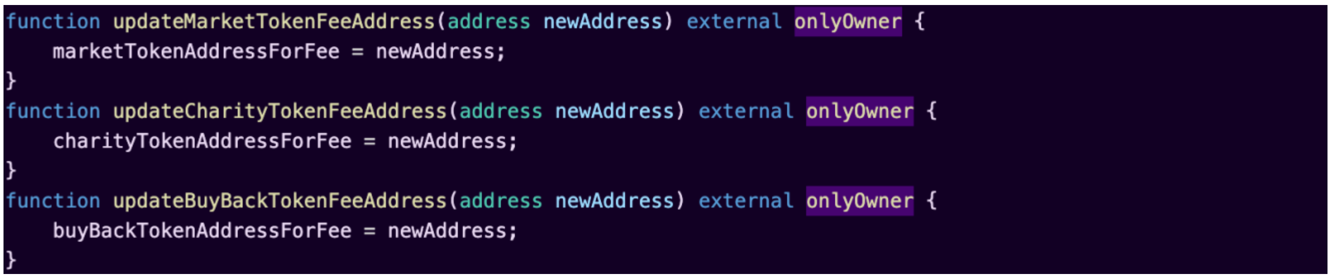
```
function updateTokenForDividend(address newAddress) external onlyOwner {
    dividendTracker.updateTokenForDividend(newAddress);
    DividendToken = newAddress;
    emit UpdateDividendToken(newAddress, address(DividendToken));
}
```

- Owner can change market, charity and buyBack BNB address.



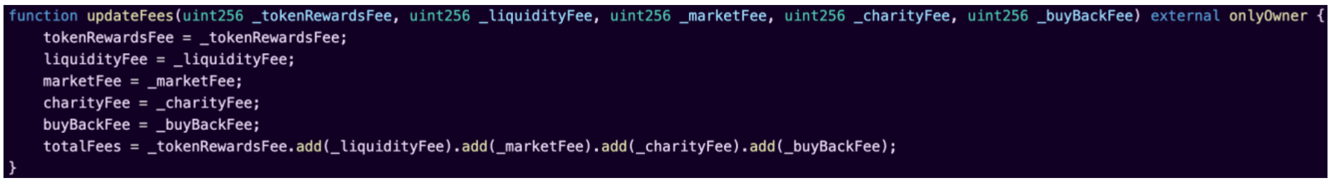
```
function updateMarketAddress(address payable newAddress) external onlyOwner {
    marketAddress = newAddress;
    _isExcludedFromFees[newAddress] = true;
}
function updateCharityAddress(address payable newAddress) external onlyOwner {
    charityAddress = newAddress;
    _isExcludedFromFees[newAddress] = true;

}
function updateBuyBackAddress(address payable newAddress) external onlyOwner {
    buyBackAddress = newAddress;
    _isExcludedFromFees[newAddress] = true;
}
```

- Owner can change market, charity and buy back token address.



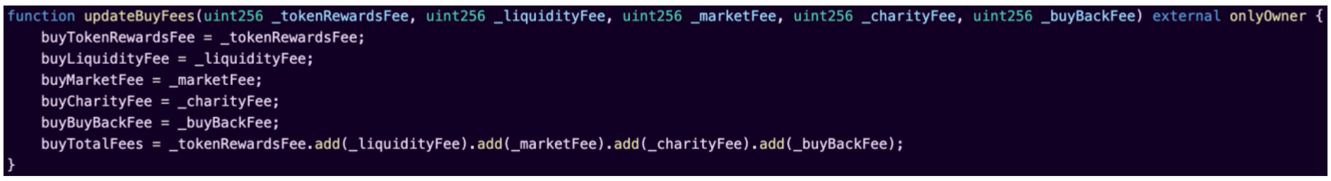
```
function updateMarketTokenFeeAddress(address newAddress) external onlyOwner {
    marketTokenAddressForFee = newAddress;
}
function updateCharityTokenFeeAddress(address newAddress) external onlyOwner {
    charityTokenAddressForFee = newAddress;
}
function updateBuyBackTokenFeeAddress(address newAddress) external onlyOwner {
    buyBackTokenAddressForFee = newAddress;
}
```

- **Owner can change token rewards, liquidity, market, charity and buy back fees.**



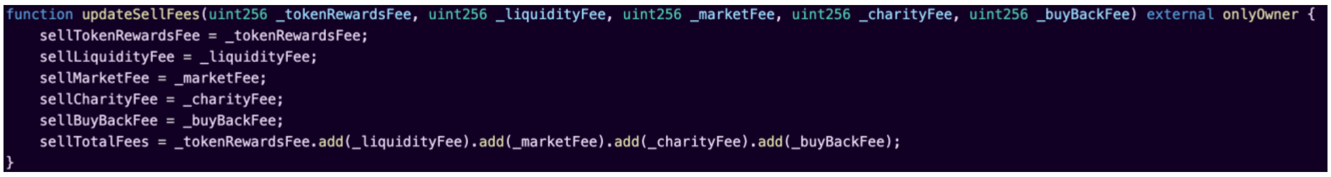
```
function updateFees(uint256 _tokenRewardsFee, uint256 _liquidityFee, uint256 _marketFee, uint256 _charityFee, uint256 _buyBackFee) external onlyOwner {
    tokenRewardsFee = _tokenRewardsFee;
    liquidityFee = _liquidityFee;
    marketFee = _marketFee;
    charityFee = _charityFee;
    buyBackFee = _buyBackFee;
    totalFees = _tokenRewardsFee.add(_liquidityFee).add(_marketFee).add(_charityFee).add(_buyBackFee);
}
```

- **Owner can change token rewards, liquidity, market, charity and buy back fees for buy.**



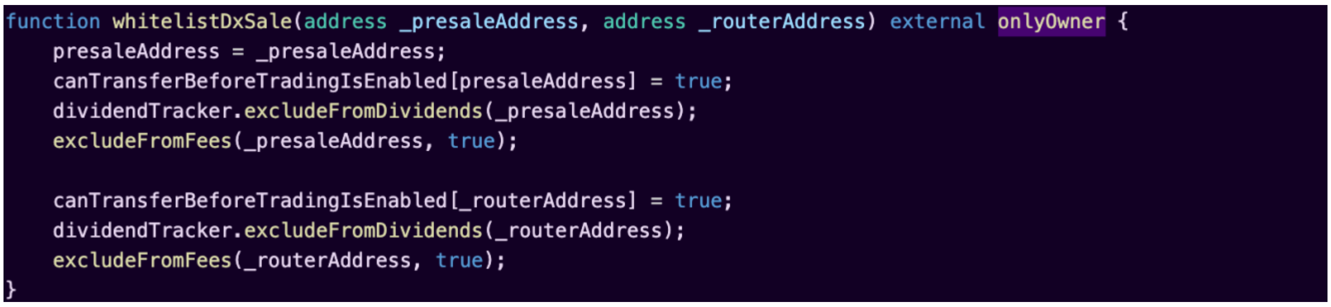
```
function updateBuyFees(uint256 _tokenRewardsFee, uint256 _liquidityFee, uint256 _marketFee, uint256 _charityFee, uint256 _buyBackFee) external onlyOwner {
    buyTokenRewardsFee = _tokenRewardsFee;
    buyLiquidityFee = _liquidityFee;
    buyMarketFee = _marketFee;
    buyCharityFee = _charityFee;
    buyBuyBackFee = _buyBackFee;
    buyTotalFees = _tokenRewardsFee.add(_liquidityFee).add(_marketFee).add(_charityFee).add(_buyBackFee);
}
```

- **Owner can change token rewards, liquidity, market, charity and buy back fees for sell.**



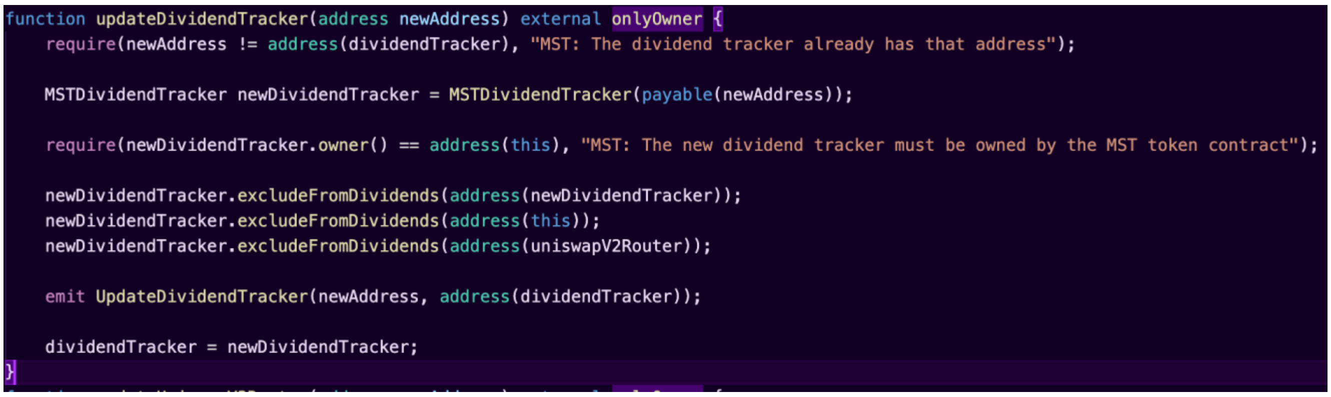
```
function updateSellFees(uint256 _tokenRewardsFee, uint256 _liquidityFee, uint256 _marketFee, uint256 _charityFee, uint256 _buyBackFee) external onlyOwner {
    sellTokenRewardsFee = _tokenRewardsFee;
    sellLiquidityFee = _liquidityFee;
    sellMarketFee = _marketFee;
    sellCharityFee = _charityFee;
    sellBuyBackFee = _buyBackFee;
    sellTotalFees = _tokenRewardsFee.add(_liquidityFee).add(_marketFee).add(_charityFee).add(_buyBackFee);
}
```

- **Owner can change presale and router address.**



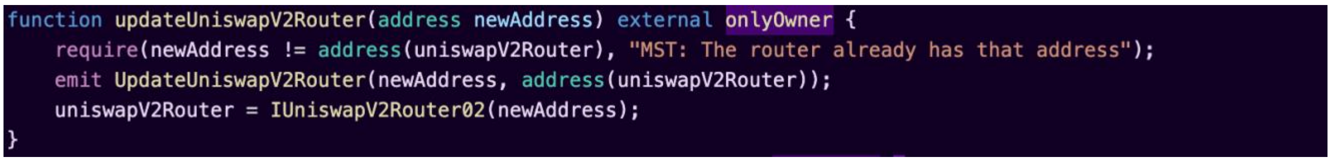
```
function whitelistDxSale(address _presaleAddress, address _routerAddress) external onlyOwner {
    presaleAddress = _presaleAddress;
    canTransferBeforeTradingIsEnabled[presaleAddress] = true;
    dividendTracker.excludeFromDividends(_presaleAddress);
    excludeFromFees(_presaleAddress, true);

    canTransferBeforeTradingIsEnabled[_routerAddress] = true;
    dividendTracker.excludeFromDividends(_routerAddress);
    excludeFromFees(_routerAddress, true);
}
```

- **Owner can change dividend tracker.**



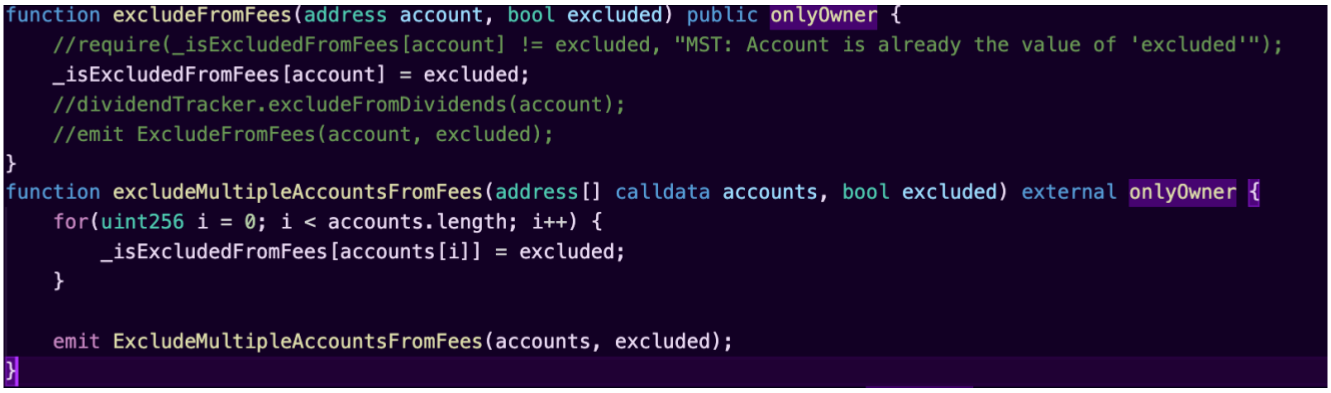
```
function updateDividendTracker(address newAddress) external onlyOwner {
    require(newAddress != address(dividendTracker), "MST: The dividend tracker already has that address");

    MSTDividendTracker newDividendTracker = MSTDividendTracker(payable(newAddress));

    require(newDividendTracker.owner() == address(this), "MST: The new dividend tracker must be owned by the MST token contract");

    newDividendTracker.excludeFromDividends(address(newDividendTracker));
    newDividendTracker.excludeFromDividends(address(this));
    newDividendTracker.excludeFromDividends(address(uniswapV2Router));

    emit UpdateDividendTracker(newAddress, address(dividendTracker));

    dividendTracker = newDividendTracker;
}
```

- **Owner can change Uniswap router.**



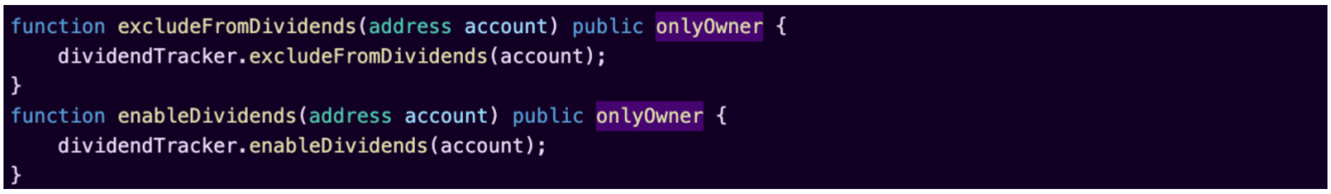
```
function updateUniswapV2Router(address newAddress) external onlyOwner {
    require(newAddress != address(uniswapV2Router), "MST: The router already has that address");
    emit UpdateUniswapV2Router(newAddress, address(uniswapV2Router));
    uniswapV2Router = IUniswapV2Router02(newAddress);
}
```

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



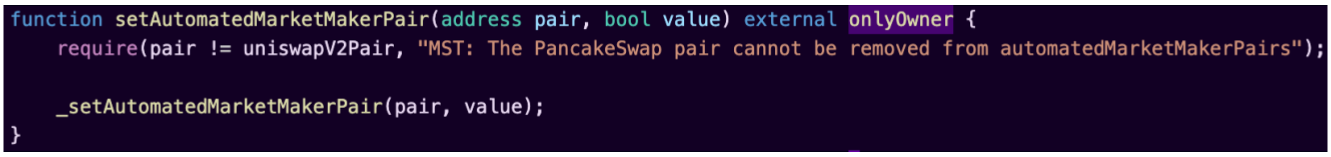
```
function excludeFromFees(address account, bool excluded) public onlyOwner {
    //require(_isExcludedFromFees[account] != excluded, "MST: Account is already the value of 'excluded'");
    _isExcludedFromFees[account] = excluded;
    //dividendTracker.excludeFromDividends(account);
    //emit ExcludeFromFees(account, excluded);
}
function excludeMultipleAccountsFromFees(address[] calldata accounts, bool excluded) external onlyOwner {
    for(uint256 i = 0; i < accounts.length; i++) {
        _isExcludedFromFees[accounts[i]] = excluded;
    }

    emit ExcludeMultipleAccountsFromFees(accounts, excluded);
}
```

- **Owner can enable in and disable from dividends.**



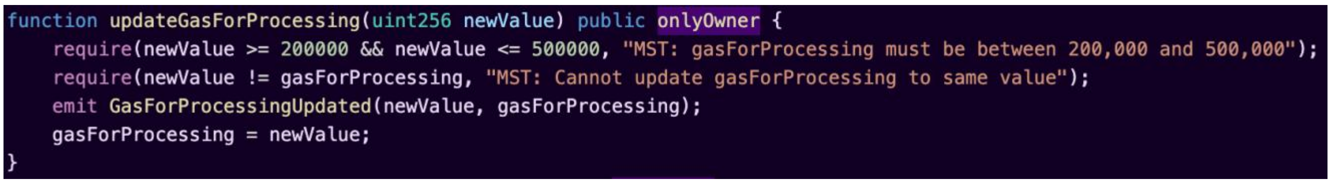
```
function excludeFromDividends(address account) public onlyOwner {
    dividendTracker.excludeFromDividends(account);
}
function enableDividends(address account) public onlyOwner {
    dividendTracker.enableDividends(account);
}
```

- **Owner can exclude and include addresses in automatedMarketMakerPairs array.**

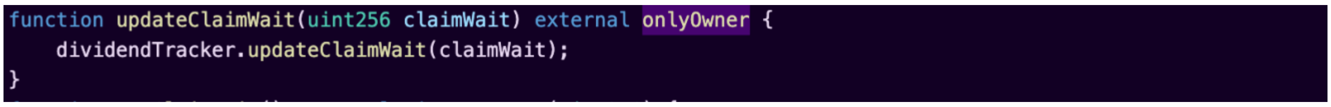


```
function setAutomatedMarketMakerPair(address pair, bool value) external onlyOwner {
    require(pair != uniswapV2Pair, "MST: The PancakeSwap pair cannot be removed from automatedMarketMakerPairs");

    _setAutomatedMarketMakerPair(pair, value);
}
```

- **Owner can change gas for processing.**


```
function updateGasForProcessing(uint256 newValue) public onlyOwner {
    require(newValue >= 200000 && newValue <= 500000, "MST: gasForProcessing must be between 200,000 and 500,000");
    require(newValue != gasForProcessing, "MST: Cannot update gasForProcessing to same value");
    emit GasForProcessingUpdated(newValue, gasForProcessing);
    gasForProcessing = newValue;
}
```

- **Owner can update claimWait value.**


```
function updateClaimWait(uint256 claimWait) external onlyOwner {
    dividendTracker.updateClaimWait(claimWait);
}
```

Conclusion

Smart contracts contain low severity issues! Liquidity pair contract's security is not checked due to out of scope. The further transfers and operations with the funds raise are not related to this particular contract.

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.