COINSCOPE

# Audit Report

# Doge Digger

January 2022

| | |
|---|---|
| Type | BEP20 |
| Network | BSC |
| Address | 0x94CBe5Aecb7E4B978E0B578EA5eF30D7db54D208 |
| Audited by | © coinscope |

# Table of Contents

# Contract Review

| | |
|---|---|
| **Contract Name** | DOGEDIGGER |
| **Compiler Version** | v0.7.4+commit.3f05b770 |
| **Optimization** | 200 runs |
| **Licence** | None |
| **Explorer** | https://bscscan.com/token/0x94CBe5Aecb7E4B978E0B578EA5eF30D7db54D208 |
| **Symbol** | DIGGER |
| **Decimals** | 8 |
| **Total Supply** | 1,000,000,000,000,000 |
| **Source** | contract.sol |
| **Domain** | dogedigger.net |

# Audit Updates

| | |
|---|---|
| **Initial Audit** | 13th January 2022 |
| **Corrected** | |

# Contract Analysis

🔴 Critical 🟠 Medium ⚪ Minor 🔵 Pass

| Severity | Code | Description |
|---|---|---|
| 🔴 | ST | Contract Owner is not able to stop or pause transactions |
| 🔵 | OCTD | Contract Owner is not able to transfer tokens from specific address |
| 🔵 | OTUT | Owner Transfer User's Tokens |
| 🔴 | ELFM | Contract Owner is not able to increase fees more than a reasonable percent (25%) |
| 🔵 | ULTW | Contract Owner is not able to increase the amount of liquidity taken by dev wallet more than a reasonable percent |
| 🔵 | MT | Contract Owner is not able to mint new tokens |
| 🔵 | BT | Contract Owner is not able to burn tokens from specific wallet |
| 🔴 | BC | Contract Owner is not able to blacklist wallets from selling |

# ST - Stop Transactions

| | |
|---|---|
| **Criticality** | critical |
| **Location** | contract.sol#L536 |

## Description

The contract owner has the authority to stop transactions for all users excluding the owner. The owner may take advantage of it by setting the `_maxTxAmount` to zero or by setting the `tradingOpen` to false

```
if(!authorizations[sender] && !authorizations[recipient]){
    require(tradingOpen,"Trading not open yet");
}

uint256 rAmount = amount.mul(rate);

if (!authorizations[sender] && recipient != address(this)  && recipient !=
address(DEAD) && recipient != pair && recipient != marketingFeeReceiver &&
recipient != devFeeReceiver  && recipient != autoLiquidityReceiver){
    uint256 heldTokens = balanceOf(recipient);
    require((heldTokens + rAmount) <= _maxWalletToken,"Total Holding is
currently limited, you can not buy that much.");}

if (sender == pair &&
    buyCooldownEnabled &&
    !isTimelockExempt[recipient]) {
    require(cooldownTimer[recipient] < block.timestamp,"buy Cooldown exists");
    cooldownTimer[recipient] = block.timestamp + cooldownTimerInterval;
}
```

## Recommendation

The contract could embody a check for not allowing setting the _maxTxAmount less than a reasonable amount. A suggested implementation could check that the maximum amount should be more than a fixed percentage of the total supply.

The contract should also prevent the owner from disabling the tradingOpen variable. Usually this variables toggles only once from false to true.

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

# ELFM - Exceed Limit Fees Manipulation

| | |
|---|---|
| **Criticality** | critical |
| **Location** | contract.sol#L614 |

## Description

The contract owner has the authority to increase over the allowed limit of 25%. The owner may take advantage of it by calling the `set_sell_multiplier` function with a high percentage value.

```
uint256 multiplier = 100;
if(isSell){
    multiplier = sellMultiplier;
}

uint256 feeAmount = rAmount.div(feeDenominator *
100).mul(totalFee).mul(multiplier);
```

## Recommendation

The contract could embody a check for the maximum acceptable value.

The team should carefully manage the private keys of the owner’s account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

# BC - Blacklisted Contracts

| | |
|---|---|
| **Criticality** | critical |
| **Location** | contract.sol#L554 |

## Description

The contract owner has the authority to stop contracts from transactions. The owner may take advantage of it by enabling the `rocketFuel` variable and setting the `maxFuelPrice` to zero simultaneously.

```
if(rocketFuel){
    require(!isSniper[sender],"Bots cant sell");
    if(tx.gasprice > maxFuelPrice && sender == pair){
        isSniper[recipient] = true;
        isDividendExempt[recipient] = true;
        emit RocketFuelCheckin(recipient, tx.gasprice);
    }
}
```

## Recommendation

The team should carefully manage the private keys of the owner’s account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

# Contract Diagnostics

● Critical ● Medium ● Minor

| Severity | Code | Description |
|---|---|---|
| ● | L01 | Public Function could be Declared External |
| ● | L02 | State Variables could be Declared Constant |
| ● | L05 | Unused State Variable |
| ● | L04 | Conformance to Solidity Naming Conventions |
| ● | L09 | Dead Code Elimination |
| ● | L07 | Missing Events Arithmetic |
| ● | L06 | Missing Events Access Control |

# L01 - Public Function could be Declared External

| | |
|---|---|
| **Criticality** | minor |
| **Location** | contract.sol#L845,L693,L687 and 8 more |

## Description

Public functions that are never called by the contract should be declared external to save gas.

```
rescueToken
cooldownEnabled
manage_blacklist
...
```

## Recommendation

Use the external attribute for functions never called from the contract

# L02 - State Variables could be Declared Constant

| | |
|---|---|
| **Criticality** | minor |
| **Location** | contract.sol#L229,L216,L394 and 3 more |

## Description

Constant state variables should be declared constant to save gas.

```
dividendsPerShareAccuracyFactor
WBNB
buybackFee
...
```

## Recommendation

Add the constant attribute to state variables that never change.

# L05 - Unused State Variable

| | |
|---|---|
| **Criticality** | minor |
| **Location** | contract.sol#L449,L47 |

## Description

There are segments that contains unused state variable.

```
MAX_SUPPLY
MAX_INT256
```

## Recommendation

Remove unused state variables.

# L04 - Conformance to Solidity Naming Conventions

| | |
|---|---|
| **Criticality** | minor |
| **Location** | contract.sol#L454,L453,L450 and 51 more |

## Description

Solidity defines a naming convention that should be followed. Rule exceptions:

- Allow constant variable name/symbol/decimals to be lowercase.
- Allow _ at the beginning of the mixed_case match for private variables and unused parameters.

```
_maxWalletToken
_maxTxAmount
rSupply
...
```

## Recommendation

Follow the Solidity naming convention.
https://docs.soliditylang.org/en/v0.4.25/style-guide.html#naming-conventions

# L09 - Dead Code Elimination

| | |
|---|---|
| **Criticality** | minor |
| **Location** | contract.sol#L63,L49,L57 and 2 more |

## Description

Functions that are not used in the contract, and make the code's size bigger.

```
sub
mul
div
...
```

## Recommendation

Remove unused functions.

# L07 - Missing Events Arithmetic

| | |
|---|---|
| **Criticality** | minor |
| **Location** | contract.sol#L877,L873,L801 and 6 more |

## Description

Detected missing events for critical arithmetic parameters. There are functions that have no event emitted, so it is difficult to track off-chain changes.

```
_maxTxAmount = rSupply.div(1000).mul(maxTXPercentage_base1000)
_maxWalletToken = rSupply.div(1000).mul(maxWallPercent_base1000)
targetLiquidity = _target
...
```

## Recommendation

Emit an event for critical parameter changes.

# L06 - Missing Events Access Control

| | |
|---|---|
| **Criticality** | minor |
| **Location** | contract.sol#L818 |

## Description

Detected missing events for critical access control parameters. There are functions that have no event emitted, so it is difficult to track off-chain changes.

```
master = _master
```

## Recommendation

Emit an event for critical parameter changes.

# Contract Functions

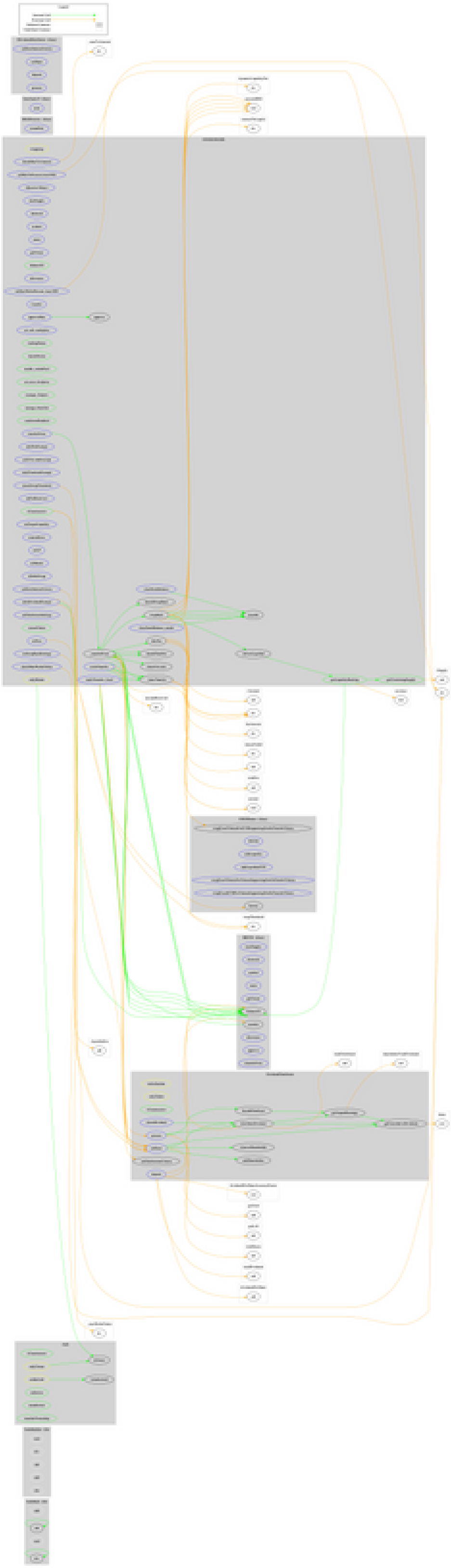
| Contract | Type | Bases | | |
|---|---|---|---|---|
| | **Function Name** | **Visibility** | **Mutability** | **Modifiers** |
| | | | | |
| **SafeMath** | Library | | | |
| | add | Internal | | |
| | sub | Internal | | |
| | sub | Internal | | |
| | mul | Internal | | |
| | div | Internal | | |
| | div | Internal | | |
| | | | | |
| **SafeMathInt** | Library | | | |
| | mul | Internal | | |
| | div | Internal | | |
| | sub | Internal | | |
| | add | Internal | | |
| | abs | Internal | | |
| | | | | |
| **IBEP20** | Interface | | | |
| | totalSupply | External | | - |
| | decimals | External | | - |
| | symbol | External | | - |
| | name | External | | - |
| | getOwner | External | | - |
| | balanceOf | External | | - |
| | transfer | External | ✓ | - |
| | allowance | External | | - |
| | approve | External | ✓ | - |
| | transferFrom | External | ✓ | - |
| | | | | |
| **Auth** | Implementation | | | |
| | <Constructor> | Public | ✓ | - |

| | | | | |
|---|---|---|---|---|
| | authorize | Public | ✓ | onlyOwner |
| | unauthorize | Public | ✓ | onlyOwner |
| | isOwner | Public | | - |
| | isAuthorized | Public | | - |
| | transferOwnership | Public | ✓ | onlyOwner |
| | | | | |
| **IDEXFactory** | Interface | | | |
| | createPair | External | ✓ | - |
| | | | | |
| **InterfaceLP** | Interface | | | |
| | sync | External | ✓ | - |
| | | | | |
| **IDEXRouter** | Interface | | | |
| | factory | External | | - |
| | WETH | External | | - |
| | addLiquidity | External | ✓ | - |
| | addLiquidityETH | External | Payable | - |
| | swapExactTokensForTokensSupportingFeeOnTransferTokens | External | ✓ | - |
| | swapExactETHForTokensSupportingFeeOnTransferTokens | External | Payable | - |
| | swapExactTokensForETHSupportingFeeOnTransferTokens | External | ✓ | - |
| | | | | |
| **IDividendDistributor** | Interface | | | |
| | setDistributionCriteria | External | ✓ | - |
| | setShare | External | ✓ | - |
| | deposit | External | Payable | - |
| | process | External | ✓ | - |
| | | | | |
| **DividendDistributor** | Implementation | IDividendDistributor | | |
| | <Constructor> | Public | ✓ | - |
| | setDistributionCriteria | External | ✓ | onlyToken |
| | setShare | External | ✓ | onlyToken |
| | deposit | External | Payable | onlyToken |

| | | | | |
|---|---|---|---|---|
| | process | External | ✓ | onlyToken |
| | shouldDistribute | Internal | | |
| | distributeDividend | Internal | ✓ | |
| | claimDividend | External | ✓ | - |
| | getUnpaidEarnings | Public | | - |
| | getCumulativeDividends | Internal | | |
| | addShareholder | Internal | ✓ | |
| | removeShareholder | Internal | ✓ | |
| | | | | |
| **DOGEDIGGER** | Implementation | IBEP20, Auth | | |
| | <Constructor> | Public | ✓ | Auth |
| | <Receive Ether> | External | Payable | - |
| | totalSupply | External | | - |
| | decimals | External | | - |
| | symbol | External | | - |
| | name | External | | - |
| | getOwner | External | | - |
| | balanceOf | Public | | - |
| | allowance | External | | - |
| | approve | Public | ✓ | - |
| | approveMax | External | ✓ | - |
| | transfer | External | ✓ | - |
| | transferFrom | External | ✓ | - |
| | _transferFrom | Internal | ✓ | |
| | _basicTransfer | Internal | ✓ | |
| | checkTxLimit | Internal | | |
| | shouldTakeFee | Internal | | |
| | takeFee | Internal | ✓ | |
| | shouldSwapBack | Internal | | |
| | clearStuckBalance | External | ✓ | authorized |
| | clearStuckBalance_sender | External | ✓ | authorized |
| | set_sell_multiplier | External | ✓ | onlyOwner |
| | tradingStatus | Public | ✓ | onlyOwner |
| | launchStatus | Public | ✓ | onlyOwner |
| | enable_rocketFuel | Public | ✓ | onlyOwner |

| | | | | |
|---|---|---|---|---|
| | set_max_fuelprice | Public | ✓ | onlyOwner |
| | manage_Snipers | Public | ✓ | onlyOwner |
| | manage_blacklist | Public | ✓ | onlyOwner |
| | cooldownEnabled | Public | ✓ | onlyOwner |
| | swapBack | Internal | ✓ | swapping |
| | setIsDividendExempt | External | ✓ | authorized |
| | setIsFeeExempt | External | ✓ | authorized |
| | setIsTxLimitExempt | External | ✓ | authorized |
| | setIsTimelockExempt | External | ✓ | authorized |
| | setFees | External | ✓ | authorized |
| | setFeeReceivers | External | ✓ | authorized |
| | setSwapBackSettings | External | ✓ | authorized |
| | setTargetLiquidity | External | ✓ | authorized |
| | manualSync | External | ✓ | - |
| | setLP | External | ✓ | onlyOwner |
| | setMaster | External | ✓ | onlyOwner |
| | isNotInSwap | External | | - |
| | checkSwapThreshold | External | | - |
| | setDistributionCriteria | External | ✓ | authorized |
| | setDistributorSettings | External | ✓ | authorized |
| | rescueToken | Public | ✓ | onlyOwner |
| | getCirculatingSupply | Public | | - |
| | getLiquidityBacking | Public | | - |
| | isOverLiquified | Public | | - |
| | checkMaxWalletToken | External | | - |
| | checkMaxTxAmount | External | | - |
| | setMaxWalletPercent_base1000 | External | ✓ | onlyOwner |
| | setMaxTxPercent_base1000 | External | ✓ | onlyOwner |
| | multiTransfer | External | ✓ | onlyOwner |
| | multiTransfer_fixed | External | ✓ | onlyOwner |

# Contract Flow



# Domain Info

| | |
|---|---|
| **Domain Name** | |
| **Registry Domain ID** | 2662437553_DOMAIN_NET-VRSN |
| **Creation Date** | 2021-12-18T17:08:05.00Z |
| **Updated Date** | 0001-01-01T00:00:00.00Z |
| **Registry Expiry Date** | |
| **Registrar WHOIS Server** | whois.namecheap.com |
| **Registrar URL** | http://www.namecheap.com |
| **Registrar** | NAMECHEAP INC |
| **Registrar IANA ID** | 1068 |

The domain has been created 26 days before the creation of the audit.

There is no public billing information, the creator is protected by the privacy settings.

# Summary

Doge Digger is a meme token. The token has a friendly and growing community. There are some functions that can be abused by the owner, like manipulating fees, stopping transactions and massively blacklisting contracts.  A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats.

# Disclaimer

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment.

Coinscope team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed.

The Coinscope team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Coinscope receive a payment to manipulate those results or change the awarding badge that we will be adding in our website.

Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token.

The Coinscope team disclaims any liability for the resulting losses.

# About Coinscope

Coinscope is the leading early coin listing, voting and auditing authority firm. The audit process is analyzing and monitoring many aspects of the project. That way, it gives the community a good sense of security using an informative report and a generic score.

Coinscope is aiming to make crypto discoverable and efficient globally. It provides all the essential tools to assist users draw their own conclusions.

The Coinscope.co team

https://www.coinscope.co