freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Reunion dog

$Reunion dog

12/02/2022

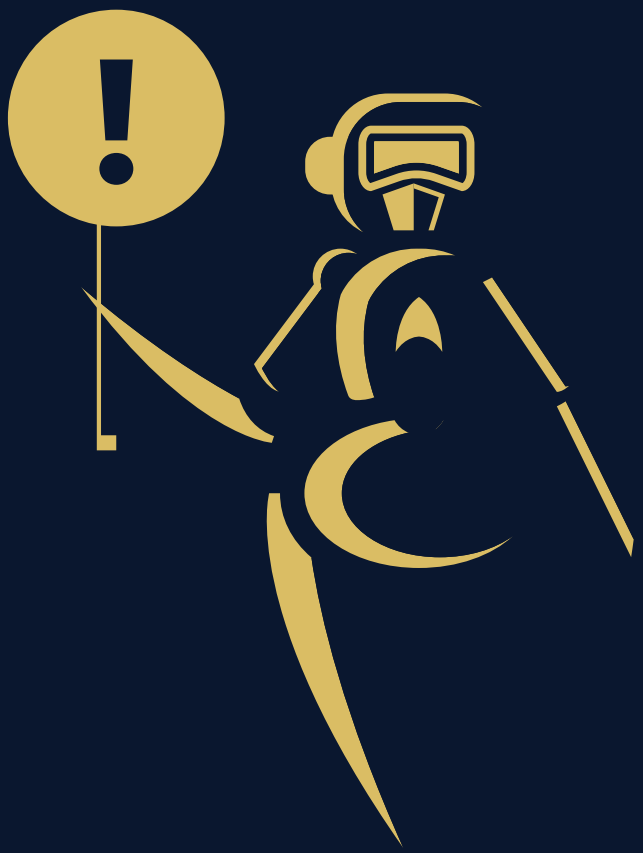
TABLE OF CONTENTS

DISCLAIMER

The information provided on this analysis document is only for general information and should not be used as a reason to invest.

FreshCoins Team will take no payment for manipulating the results of this audit.

The score and the result will stay on this project page information on our website https://freshcoins.io
FreshCoins Team does not guarantees that a project will not sell off team supply, or any other scam strategy (RUG or Honeypot etc)



INTRODUCTION

FreshCoins (Consultant) was contracted by Reunion dog (Customer) to conduct a Smart Contract Code Review and Security Analysis.

0xf16e42B93b7cD862422C36e8642BB57Db8B95F7C

Network: Binance Smart Chain (BNB)

This report presents the findings of the security assessment of Customer's smart contract and its code review conducted on 12/02/2022

AUDIT OVERVIEW

96

Security Score

94

Static Scan
Automatic scanning for common vulnerabilities

98

ERC Scan
Automatic checks for ERC's conformance

0 High

0 Medium

1 Low

0 Optimizations

0 Informational

No.	Issue description	Checking Status
1	Compiler Errors / Warnings	Passed
2	Reentrancy and Cross-function	Passed
3	Front running	Passed
4	Timestamp dependence	Passed
5	Integer Overflow and Underflow	Passed
6	Reverted DoS	Passed
7	DoS with block gas limit	Low
8	Methods execution permissions	Passed
9	Exchange rate impact	Passed
10	Malicious Event	Passed
11	Scoping and Declarations	Passed
12	Uninitialized storage pointers	Passed
13	Design Logic	Passed
14	Safe Zeppelin module	Passed

OWNER PRIVILEGES

Contract owner can't mint tokens after initial contract deploy

Contract owner can exclude wallet from transactions

```
function blacklistAddress(address account, bool value) external onlyOwner{
        _isBlacklisted[account] = value;
}
```

Contract owner can exclude wallet from dividends

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

Contract owner can exclude/include wallet(s) from fees

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

Contract owner can renounce ownership

```
function renounceOwnership() public virtual onlyOwner {
    emit OwnershipTransferred(_owner, address(0));
    _owner = address(0);
}
```

Contract owner can transfer ownership

```
function transferOwnership(address newOwner) public virtual onlyOwner {
    require(newOwner != address(0), "Ownable: new owner is the zero address");
    emit OwnershipTransferred(_owner, newOwner);
    _owner = newOwner;
}
```

Contract owner can change the fees up to 100%

```
function setDOGERewardsFee(uint256 value) external onlyOwner{
        DOGERewardsFee = value;
        totalFees = DOGERewardsFee.add(liquidityFee).add(marketingFee).add(burnFee).add(buybackFee);
}

function setLiquiditFee(uint256 value) external onlyOwner{
      liquidityFee = value;
      totalFees = DOGERewardsFee.add(liquidityFee).add(marketingFee).add(burnFee).add(buybackFee);
}

function setMarketingFee(uint256 value) external onlyOwner{
      marketingFee = value;
      totalFees = DOGERewardsFee.add(liquidityFee).add(marketingFee).add(burnFee).add(buybackFee);
}

function setBurnFee(uint256 value) external onlyOwner{
      burnFee = value;
      totalFees = DOGERewardsFee.add(liquidityFee).add(marketingFee).add(burnFee).add(buybackFee);
}

function setBuyBackFee(uint256 value) external onlyOwner{
      buybackFee = value;
      totalFees = DOGERewardsFee.add(liquidityFee).add(marketingFee).add(burnFee).add(buybackFee);
}
```

Contract owner can change _liquidProviderAddress, _marketingWalletAddress, _buyBackWalletAddress addresses

Current values:

_liquidProviderAddress: 0x8afcf3cfcc9ca1eca6d2d87d9db89a98247bc027

_marketingWalletAddress: 0x8b92e7f726f5b79316ff65003be18138f69d3228

_buyBackWalletAddress: 0xde18b196416b079b4303e5464b70dc804231f63d

```
function setLiquidProviderWallet(address payable wallet) external onlyOwner{
      _liquidProviderAddress = wallet;
}

function setMarketingWallet(address payable wallet) external onlyOwner{
      _marketingWalletAddress = wallet;
}

function setBuyBackWallet(address payable wallet) external onlyOwner{
      _buyBackWalletAddress = wallet;
}
```

CONCLUSION AND ANALYSIS

Smart Contracts within the scope were manually reviewed and analyzed with static tools.

Audit report overview contains all found security vulnerabilities and other issues in the reviewed code.

Found 1 LOW issue during the first review.

TOKEN DETAILS

Details

Buy fees:	17%
Sell fees:	17%
Max TX:	N/A
Max Sell:	N/A

Honeypot Risk

Ownership:	Owned
Blacklist:	Detected
Modify Max TX:	Not detected
Modify Max Sell:	Not detected
Disable Trading:	Not detected

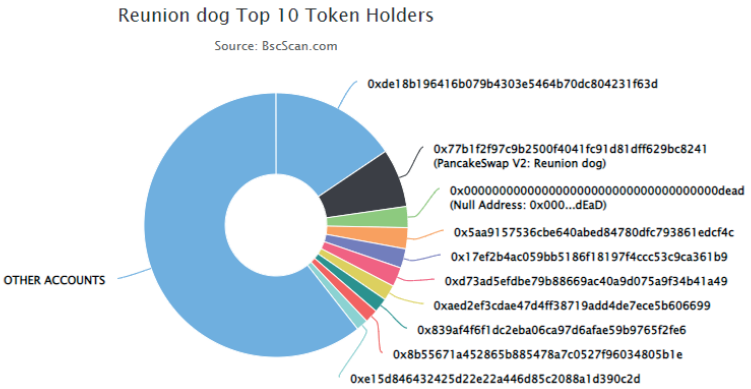
Rug Pull Risk

Liquidity:	N/A
Holders:	Clean

REUNION DOG TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

The top 10 holders collectively own 39.40% (394,001,972,773,067.00 Tokens) of Reunion dog

Token Total Supply: 1,000,000,000,000,000.00 Token | Total Token Holders: 830



(A total of 394,001,972,773,067.00 tokens held by the top 10 accounts from the total supply of 1,000,000,000,000,000.00 token)

Rank	Address	Quantity (Token)	Percentage
1	0xde18b196416b079b4303e5464b70dc804231f63d	155,719,912,555,502.125465585925180646	15.5720%
2	PancakeSwap V2: Reunion dog	71,704,372,400,295.795012954798500931	7.1704%
3	Null Address: 0x000...dEaD	25,799,473,269,716.636733821524131852	2.5799%
4	0x5aa9157536cbe640abed84780dfc793861edcf4c	25,773,216,062,187.829024514171122315	2.5773%
5	0x17ef2b4ac059bb5186f18197f4ccc53c9ca361b9	23,678,655,000,000	2.3679%
6	0xd73ad5efdbe79b88669ac40a9d075a9f34b41a49	23,657,365,500,000	2.3657%
7	0xaed2ef3cdae47d4ff38719add4de7ece5b606699	18,925,892,400,000	1.8926%
8	0x839af4f6f1dc2eba06ca97d6afae59b9765f2fe6	17,724,616,840,289.185057803255110263	1.7725%
9	0x8b55671a452865b885478a7c0527f96034805b1e	16,629,837,885,328.510701786708649978	1.6630%
10	0xe15d846432425d22e22a446d85c2088a1d390c2d	14,388,630,859,746.771778348573794682	1.4389%

TECHNICAL DISCLAIMER

Smart contracts are deployed and executed on the blockchain platform. The platform, its programming language, and other software related to the smart contract can have its vulnerabilities that can lead to hacks. The audit can't guarantee the explicit security of the audited project / smart contract.