freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Chonker Inu

$CHONKY

05/06/2023

TOKEN OVERVIEW

Fees

- Buy fees: 2%
- Sell fees: 2%

Fees privileges

- Can change buy fees up to 5% and sell fees up to 5%

Ownership

- Owned

Minting

- No mint function

Max Tx Amount / Max Wallet Amount

- Can change max tx amount and max wallet amount (with threshold)

Blacklist

- Blacklist function not detected

Other privileges

- Can exclude / include from fees

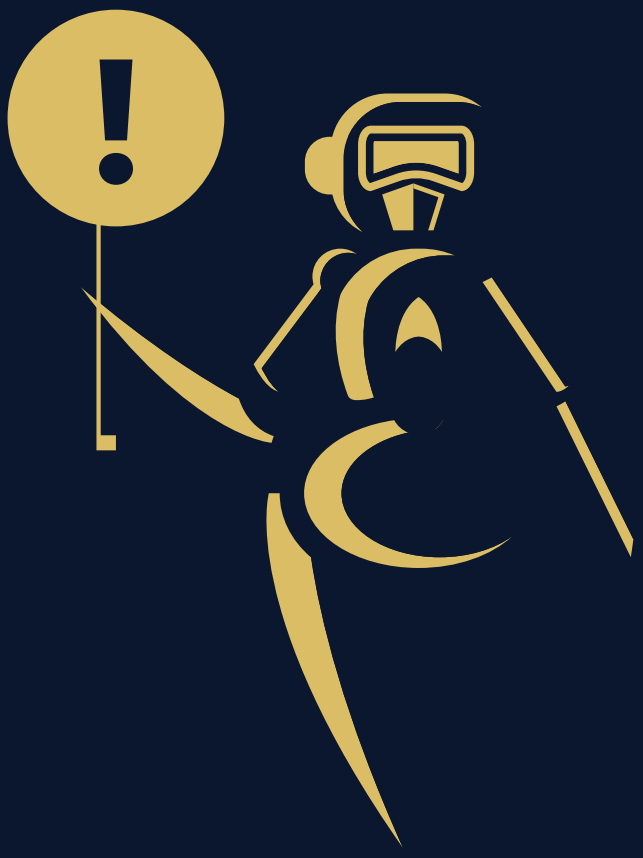
TABLE OF CONTENTS

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 Chonker Inu (Customer) to conduct a Smart Contract Code Review and Security Analysis.

0xd00138ccf5F0b55351646D5C5FE2D95Ea32A63b9

Network: Binance Smart Chain (BSC)

This report presents the findings of the security assessment of Customer's smart contract and its code review conducted on 05/06/2023

WEBSITE DIAGNOSTIC

https://chonkerinu.com/

0-49 50-89 90-100

Performance Accessibility Best Practices SEO Progressive Web App

Socials

Twitter

https://twitter.com/chonkerinu

Telegram

https://t.me/ChonkerInu

AUDIT OVERVIEW

91

Security Score

98

Static Scan
Automatic scanning for common vulnerabilities

86

ERC Scan
Automatic checks for ERC's conformance

0 High

2 Medium

0 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	Passed
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't exclude an address from transactions
- Contract owner can exclude/include wallet from tax

```
function excludeFromFees(address account, bool excluded) public onlyOwner {
    _isExcludedFromFees[account] = excluded;
    emit ExcludeFromFees(account, excluded);
}
```

- Contract owner can exclude/include wallet from tx limitations

```
function excludeFromMaxTransaction(address updAds, bool isEx) public onlyOwner {
    _isExcludedMaxTransactionAmount[updAds] = isEx;
}
```

- Contract owner can change swap settings

```
function updateSwapEnabled(bool enabled) external onlyOwner(){
    swapEnabled = enabled;
}

function updateSwapTokensAtAmount(uint256 newAmount) external onlyOwner returns (bool){
    require(newAmount >= totalSupply() * 1 / 100000, "Swap amount cannot be lower than 0.001% total
supply.");
    require(newAmount <= totalSupply() * 5 / 1000, "Swap amount cannot be higher than 0.5% total
supply.");
    swapTokensAtAmount = newAmount;
    return true;
}
```

- Contract owner can change marketingWallet address

Current value:

marketingWallet: 0x1bb1cfe566ffa0d3c5cda05f14ddfb2f88801a06

```
function updateMarketingWallet(address newMarketingWallet) external onlyOwner {
    emit marketingWalletUpdated(newMarketingWallet, marketingWallet);
    marketingWallet = newMarketingWallet;
}
```

Contract owner can change buy fees up to 5% and sell fees up to 5%

```
function updateFees(uint256 _marketingFeeBuy, uint256 _liquidityFeeBuy, uint256 _burnFeeBuy, uint256
_marketingFeeSell, uint256 _liquidityFeeSell, uint256 _burnFeeSell) external onlyOwner {
        buyMarketingFee = _marketingFeeBuy;
        buyLiquidityFee = _liquidityFeeBuy;
        buyBurnFee = _burnFeeBuy;
        buyTotalFees = buyMarketingFee + buyLiquidityFee + buyBurnFee;
        sellMarketingFee = _marketingFeeSell;
        sellLiquidityFee = _liquidityFeeSell;
        sellBurnFee = _burnFeeSell;
        sellTotalFees = sellMarketingFee + sellLiquidityFee + sellBurnFee;
        require(buyTotalFees <= 500, "Must keep fees at 5% or less");
        require(sellTotalFees <= 500, "Must keep fees at 5% or less");
    }
```

Contract owner has ability to retrieve any token held by the contract

(native tokens excluded)

```
function transferForeignToken(address _token, address _to) external onlyOwner returns (bool _sent) {
        require(_token != address(this), "Can't withdraw native tokens");
        uint256 _contractBalance = IERC20(_token).balanceOf(address(this));
        _sent = IERC20(_token).transfer(_to, _contractBalance);
        emit TransferForeignToken(_token, _contractBalance);
    }
```

Contract owner can change max tx limitations and max wallet limitations

(with threshold)

```
function updateMaxTxnAmount(uint256 newNum) external onlyOwner {
        require(newNum >= (totalSupply() * 25 / 10000)/1e18, "Cannot set maxTransactionAmount lower than
0.25%");
        maxTransactionAmount = newNum * (10**18);
    }

    function updateMaxWalletAmount(uint256 newNum) external onlyOwner {
        require(newNum >= (totalSupply() * 25 / 10000)/1e18, "Cannot set maxWallet lower than 0.25%");
        maxWallet = newNum * (10**18);
    }
```

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 renounce ownership

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

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. The risk can be prevented by temporarily locking the contract or renouncing ownership.

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 no HIGH issues during the first review.

TOKEN DETAILS

Details

Buy fees:	2%
Sell fees:	2%
Max TX:	420,000,000
Max Sell:	N/A

Honeypot Risk

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

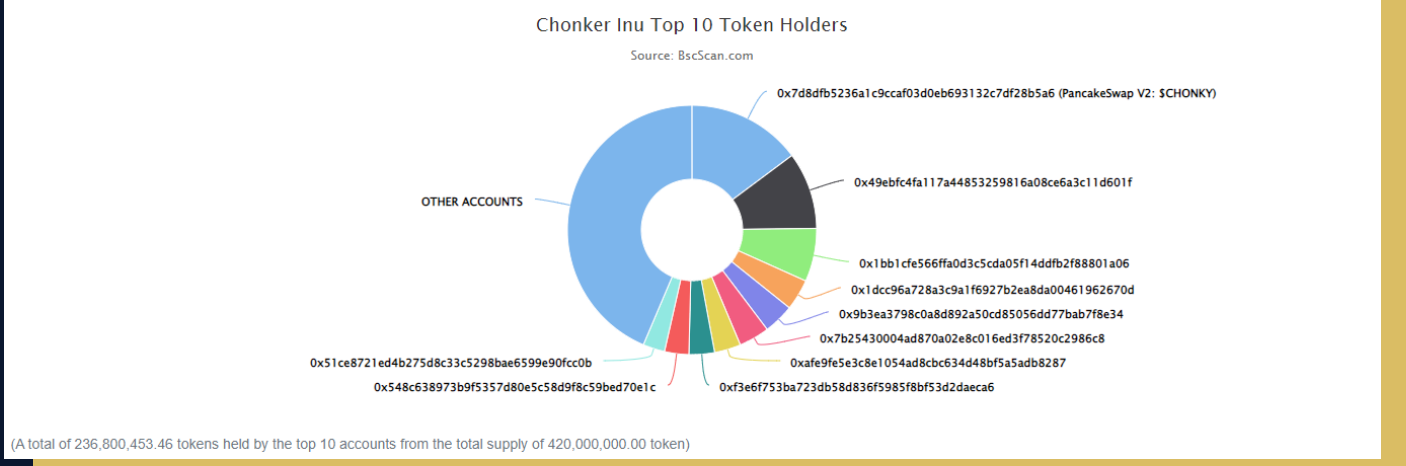
Rug Pull Risk

Liquidity:	We haven't detected locked liquidity or supply
Holders:	Clean

CHONKY TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

The top 10 holders collectively own 56.38% (236,800,453.46 Tokens) of Chonker Inu

Token Total Supply: 420,000,000.00 Token | Total Token Holders: 52



(A total of 236,800,453.46 tokens held by the top 10 accounts from the total supply of 420,000,000.00 token)

Rank	Address	Quantity (Token)	Percentage
1	PancakeSwap V2: $CHONKY	62,223,680.248590774083803342	14.8152%
2	0x49ebfc4fa117a44853259816a08ce6a3c11d601f	42,000,000	10.0000%
3	0x1bb1cfe566ffa0d3c5cda05f14ddfb2f88801a06	28,980,000	6.9000%
4	0x1dcc96a728a3c9a1f6927b2ea8da00461962670d	16,884,327.728186221795043897	4.0201%
5	0x9b3ea3798c0a8d892a50cd85056dd77bab7f8e34	16,660,000.000000000020919953	3.9667%
6	0x7b25430004ad870a02e8c016ed3f78520c2986c8	16,660,000.000000000000336591	3.9667%
7	0xafe9fe5e3c8e1054ad8cbc634d48bf5a5adb8287	14,404,669.551566821930967057	3.4297%
8	0xf3e6f753ba723db58d836f5985f8bf53d2daeca6	13,839,082.313137814504959391	3.2950%
9	0x548c638973b9f5357d80e5c58d9f8c59bed70e1c	13,241,380.821352592609626001	3.1527%
10	0x51ce8721ed4b275d8c33c5298bae6599e90fcc0b	11,907,312.792468477701350334	2.8351%

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.