freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Jok

$JOK

11/08/2023

TOKEN OVERVIEW

Fees

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

Fees privileges

- Can change fees up to 10%

Ownership

- Owned

Minting

- No mint function

Max Tx Amount / Max Wallet Amount

- Can’t change max tx amount and / or max wallet amount

Blacklist

- Blacklist function not detected

Other privileges

- Can exclude / include from fees
- Contract owner has to call enableTrading function to enable trade

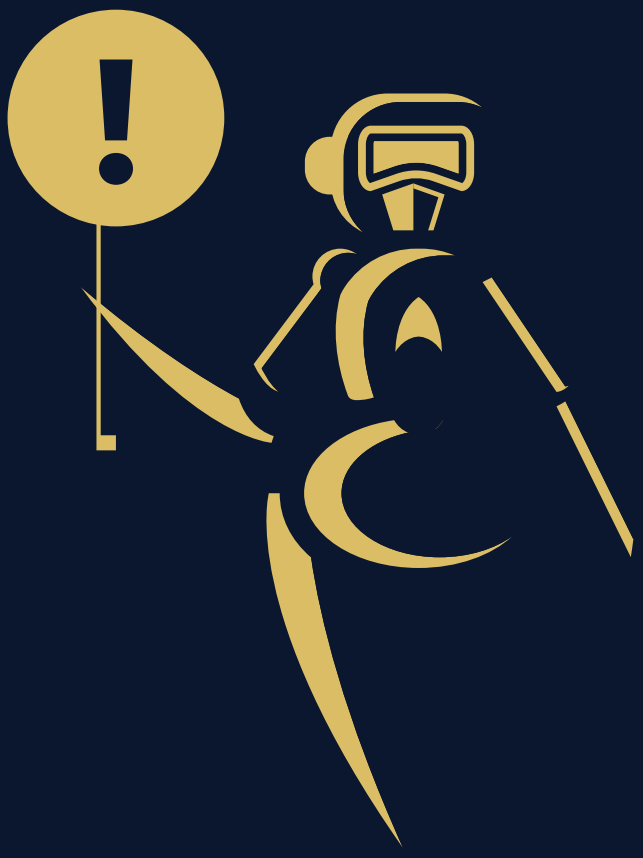
TABLE OF CONTENTS

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

0xA728Aa2De568766E2Fa4544Ec7A77f79c0bf9F97

Network: Ethereum (ETH)

This report presents the findings of the security assessment of Customer's smart contract and its code review conducted on 11/08/2023

WEBSITE DIAGNOSTIC

https://www.jokinthebox.com/

0-49 50-89 90-100

Performance	Accessibility	Best Practices	SEO	Progressive Web App
95	94	95	94	NA

Socials

Twitter

https://twitter.com/JokInTheBoxETH

Telegram

https://t.me/JokInTheBox_AI_Labs_Portal

AUDIT OVERVIEW

82

Security Score

98

Static Scan
Automatic scanning for common vulnerabilities

81

ERC Scan
Automatic checks for ERC's conformance

1 High

0 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(s) from tax

```
function excludeFromFees(
      address account,
      bool excluded
   ) external onlyOwner {
      require(
         _isExcludedFromFees[account] != excluded,
         "Account is already the value of 'excluded'"
      );
      _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 has to call enableTrading function to enable trade

Note that any wallet excluded from fees can trade even if trading is disabled

```
function enableTrading() external onlyOwner {
      require(!tradingEnabled, "Trading already enabled.");
      tradingEnabled = true;
      swapEnabled = true;
   }

... line 212 in transfer function

require(
         tradingEnabled ||
            _isExcludedFromFees[from] ||
            _isExcludedFromFees[to],
         "Trading not yet enabled!"
      );
```

Contract owner can change fees up to 10%

```
function updateBuyFees(
        uint256 _liquidityFeeOnBuy,
        uint256 _companyFeeOnBuy
    ) external onlyOwner {
        liquidityFeeOnBuy = _liquidityFeeOnBuy;
        companyFeeOnBuy = _companyFeeOnBuy;

        _totalFeesOnBuy = liquidityFeeOnBuy + companyFeeOnBuy;

        require(
            _totalFeesOnBuy + _totalFeesOnSell <= 10,
            "Total Fees cannot exceed the maximum"
        );

        emit UpdateBuyFees(liquidityFeeOnBuy, companyFeeOnBuy);
    }

    function updateSellFees(
        uint256 _liquidityFeeOnSell,
        uint256 _companyFeeOnSell
    ) external onlyOwner {
        liquidityFeeOnSell = _liquidityFeeOnSell;
        companyFeeOnSell = _companyFeeOnSell;

        _totalFeesOnSell = liquidityFeeOnSell + companyFeeOnSell;

        require(
            _totalFeesOnBuy + _totalFeesOnSell <= 10,
            "Total Fees cannot exceed the maximum"
        );

        emit UpdateSellFees(liquidityFeeOnSell, companyFeeOnSell);
    }
```

Contract owner can change liquidityWallet and companyWallet addresses

Current values:

liquidityWallet: 0x9d438A53b2e4E9d453331Da806EC4aE31eba1A0A

companyWallet: 0x9d438A53b2e4E9d453331Da806EC4aE31eba1A0A

```
function changecompanyWallet(address _companyWallet) external onlyOwner {
        require(
            _companyWallet != companyWallet,
            "company wallet is already that address"
        );
        require(
            _companyWallet != address(0),
            "company wallet cannot be the zero address"
        );
        companyWallet = _companyWallet;

        emit companyWalletChanged(companyWallet);
    }
```

```
function changeLiquidityWallet(
        address _liquidityWallet
    ) external onlyOwner {
        require(
            _liquidityWallet != liquidityWallet,
            "company wallet is already that address"
        );
        require(
            _liquidityWallet != address(0),
            "company wallet cannot be the zero address"
        );
        liquidityWallet = _liquidityWallet;

        emit LiquidityWalletChanged(liquidityWallet);
}
```

Contract owner can change swap settings

```
function setSwapEnabled(bool _enabled) external onlyOwner {
        require(swapEnabled != _enabled, "swapEnabled already at this state.");
        swapEnabled = _enabled;
    }

function setSwapTokensAtAmount(uint256 newAmount) external onlyOwner {
        require(
            newAmount > totalSupply() / 1_000_000,
            "SwapTokensAtAmount must be greater than 0.0001% of total supply"
        );
        swapTokensAtAmount = newAmount;

        emit SwapTokensAtAmountUpdated(swapTokensAtAmount);
}
```

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

Native tokens excluded

```
function claimStuckTokens(address token) external onlyOwner {
        require(
            token != address(this),
            "Owner cannot claim contract's balance of its own tokens"
        );
        if (token == address(0x0)) {
            payable(msg.sender).sendValue(address(this).balance);
            return;
        }
        IERC20 ERC20token = IERC20(token);
        uint256 balance = ERC20token.balanceOf(address(this));
        ERC20token.transfer(msg.sender, balance);
}
```

Contract owner can transfer ownership

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

function _transferOwnership(address newOwner) internal virtual {
    address oldOwner = _owner;
    _owner = newOwner;
    emit OwnershipTransferred(oldOwner, newOwner);
}
```

Contract owner can renounce ownership

```
function renounceOwnership() public virtual onlyOwner {
    _transferOwnership(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 1 HIGH issues during the first review.

TOKEN DETAILS

Details

Buy fees:	2%
Sell fees:	2%
Max TX:	N/A
Max Sell:	N/A

Honeypot Risk

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

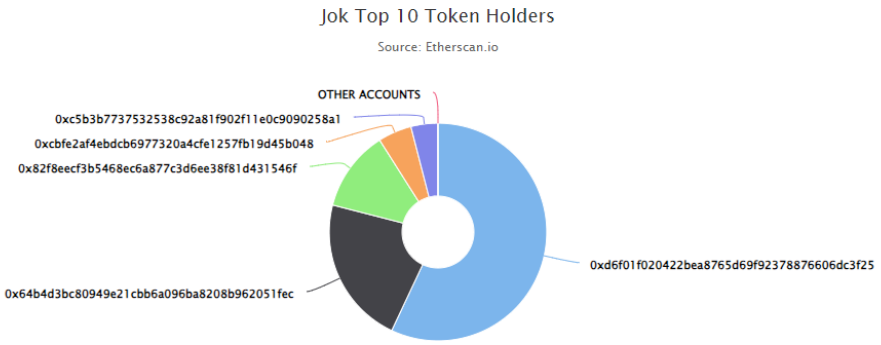
Rug Pull Risk

Liquidity:	N/A
Holders:	100% unlocked tokens

JOK TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

The top 10 holders collectively own 100.00% (777,000,000,000.00 Tokens) of Jok

Token Total Supply: 777,000,000,000.00 Token | Total Token Holders: 5



(A total of 777,000,000,000.00 tokens held by the top 10 accounts from the total supply of 777,000,000,000.00 token)

Rank	Address	Quantity (Token)	Percentage
1	0xd6f01f...06Dc3f25	442,890,000,000	57.0000%
2	0x64B4d3...62051FeC	170,940,000,000	22.0000%
3	0x82f8Ee...d431546f	93,240,000,000	12.0000%
4	0xCBFe2A...9d45B048	38,850,000,000	5.0000%
5	0xC5B3b7...090258a1	31,080,000,000	4.0000%

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.