freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

LINK IN

$LINK-IN

21/09/2022

TOKEN OVERVIEW

Fees

• Buy fees:	8%
• Sell fees:	8%

Fees privileges

• Can set buy fees up to 10% and sell fees up to 25%

Ownership

• Owned

Minting

• No mint function

Max Tx Amount / Max Wallet Amount

• Can set max tx amount, max sell amount and/or wallet limitations with threshold

Blacklist

• No blacklist function

Other privileges

• Can exclude from fees and dividends

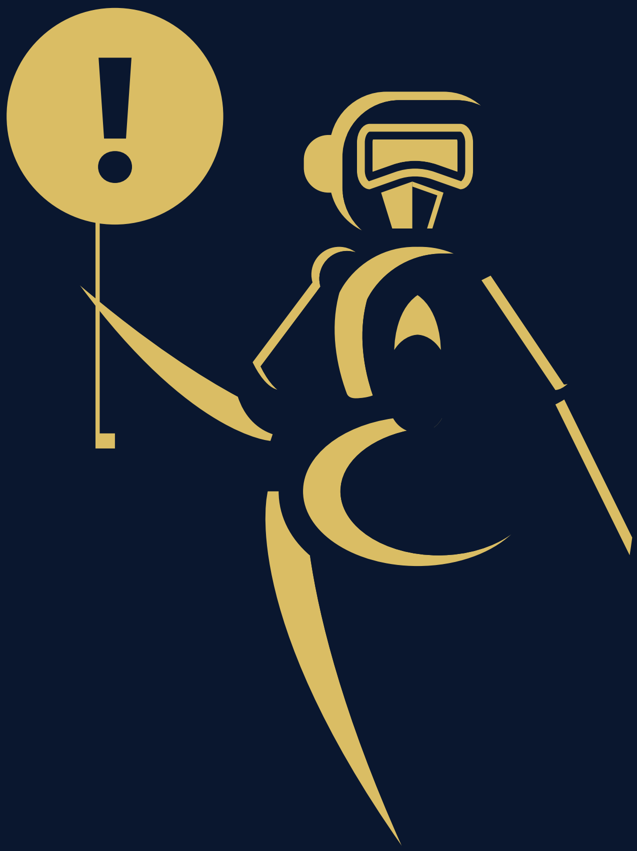
TABLE OF CONTENTS

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

0xeDeC1B47dAE0DD5631ceb8Ea1095cDc8C491A6f5

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 21/09/2022

AUDIT OVERVIEW

92

Security Score
AUDIT: PASS

92 **Static Scan**
Automatic scanning for common vulnerabilities

92 **ERC Scan**
Automatic checks for ERC's conformance

0	High
1	Medium
5	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't exclude an address from transactions

- Contract owner has to call launch() function to enable trade

```
function launch() external onlyOwner { // Used to initialize trading
      require (_launchedAt == 0, "Already launched");
      _tradingEnabled = true;
      _launchedAt = block.timestamp;
      emit Launch(block.timestamp);
}
```

- Contract owner can exclude/include wallet from fees

```
function setIsFeeExempt(address holder, bool exempt) external onlyOwner { // Set Included (false) or Exclud-
ed (true) wallets from fees
      require(holder != address(this) && holder != pair, "Cannot include pair or token contract");
      _excludeFromFees[holder] = exempt;
      emit SetIsFeeExempt(holder, exempt, block.timestamp);
}
```

- Contract owner can exclude/include wallet from dividends

```
function setIsDividendExempt(address holder, bool exempt) external onlyOwner { // Set Included (false) or
Excluded (true) wallets from rewards
      require(holder != address(this) && holder != pair, "Cannot include pair or token contract");
      _excludeFromRewards[holder] = exempt;
      if(exempt){
         distributor.setShare(holder, 0);
      }else{
         distributor.setShare(holder, _balances[holder]);
      }
      emit SetIsDividendExempt(holder, exempt, block.timestamp);
}
```

- Contract owner can change max buy and max sell amount (with threshold)

```
function setMaxBuyAmount(uint256 percentageBase100) external onlyOwner { // Owner can set limits -
Minimum allowed is 1%
      require (percentageBase100 > 0, "Cannot set 0 percentage");
      _maxBuyAmount = (_totalSupply * percentageBase100) / 100;
      emit SetMaxBuyAmount(percentageBase100, block.timestamp);
}

function setMaxSellAmount(uint256 percentageBase100) external onlyOwner { // Owner can set limits -
Minimum allowed is 1%
      require (percentageBase100 > 0, "Cannot set 0 percentage");
      _maxSellAmount = (_totalSupply * percentageBase100) / 100;
      emit SetMaxSellAmount(percentageBase100, block.timestamp);
}
```

● Contract owner can change max wallet limitations (with threshold)

```
function setMaxWalletSize(uint256 percentageBase100) external onlyOwner { // Owner can set limits - 
Minimum allowed is 1%
        require (percentageBase100 > 0, "Cannot set 0 percentage");
        _maxWalletSize = (_totalSupply * percentageBase100) / 100;
        emit SetMaxWalletSize(percentageBase100, block.timestamp);
}
```

● Contract owner can change buy fees up to 10% and sell fees up to 25%

```
function setSellFees(uint256 marketing, uint256 dev, uint256 rewards, uint256 liquidity) external onlyOwner {
// Owner can set sell fees - Maximum allowed is 25%
        require(_tradingEnabled,"Trading not enabled");
        require(marketing + dev + rewards + liquidity <= 25, "Cannot set Sell Fees higher than 25%");
        _sellMarketingFee = marketing;
        _sellDevelopmentFee = dev;
        _sellRewardsFee = rewards;
        _sellLiquidityFee = liquidity;
        _sellTotalFee = marketing + dev + rewards + liquidity;
        emit SetSellFees(marketing,dev,rewards,liquidity,block.timestamp);
}

function setBuyFees(uint256 marketing, uint256 dev, uint256 rewards, uint256 liquidity) external onlyOwner {
// Owner can set buy fees - Maximum allowed is 20%
        require(_tradingEnabled,"Trading not enabled");
        require(marketing + dev + rewards + liquidity <= 10, "Cannot set Buy Fees higher than 20%");
        _buyMarketingFee = marketing;
        _buyDevelopmentFee = dev;
        _buyRewardsFee = rewards;
        _buyLiquidityFee = liquidity;
        _buyTotalFee = marketing + dev + rewards + liquidity;
        emit SetBuyFees(marketing,dev,rewards,liquidity,block.timestamp);
}
```

● Contract owner can change devFeeReceiver and marketingFeeReceiver addresses

Default values:

devFeeReceiver : 0xEa3fB8Fe3068373EA81b82e8bec9F8768Fd316D6

marketingFeeReceiver : 0xF6b5e4027418f7872Fa34F6ec01146Db80e84244

```
function setDevWallet(address devWallet) external onlyOwner { // Owner can set new Dev Fee receiver
        require(devFeeReceiver != devWallet,"This address is already DevWallet");
        address oldWallet = devFeeReceiver;
        devFeeReceiver = devWallet;
        emit SetDevWallet(oldWallet, devWallet);
}

function setMarketingWallet(address marketingWallet) external onlyOwner { // Owner can set new Marketing
Fee receiver
        require(marketingFeeReceiver != marketingWallet,"This address is already MarketingWallet");
        address oldWallet = marketingFeeReceiver;
        marketingFeeReceiver = marketingWallet;
        emit SetMarketingWallet(oldWallet, marketingWallet);
}
```

Contract owner can change swap settings

```
function setSwapDetails(bool enabled, uint256 threshold, uint256 maxSwap) external onlyOwner { // Set
SwapBack details
      require(threshold > 0);
      _swapEnabled = enabled;
      swapThreshold = threshold * 10 ** _decimals; // Add token amount without decimals
      _maxSwapSize = maxSwap * 10 ** _decimals; // Add token amount without decimals
      emit SetSwapDetails(threshold, maxSwap);
}
```

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;
}
```

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:	8%
Sell fees:	8%
Max TX:	30,000,000
Max Sell:	10,000,000

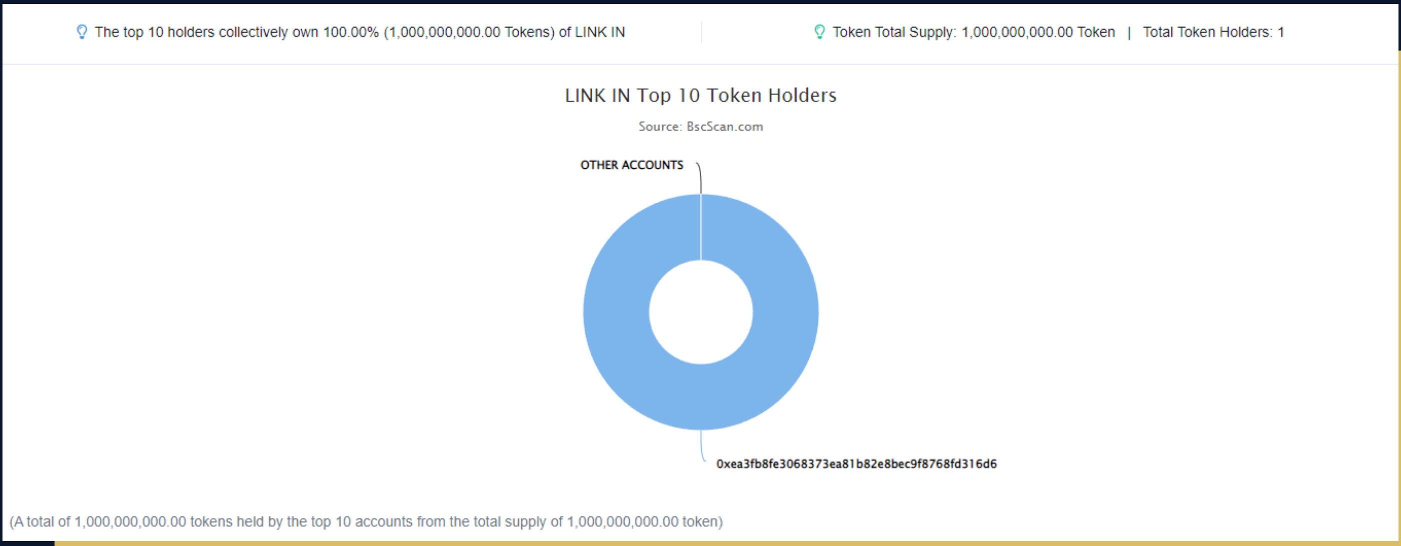
Honeypot Risk

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

Others

Liquidity:	N/A
Holders:	Clean

LINK IN TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS



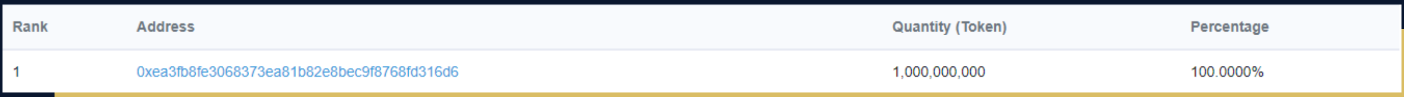
The top 10 holders collectively own 100.00% (1,000,000,000.00 Tokens) of LINK IN

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

LINK IN Top 10 Token Holders

Source: BscScan.com

(A total of 1,000,000,000.00 tokens held by the top 10 accounts from the total supply of 1,000,000,000.00 token)



Rank	Address	Quantity (Token)	Percentage
1	0xea3fb8fe3068373ea81b82e8bec9f8768fd316d6	1,000,000,000	100.0000%

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.