freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Lazy Coin (LAZYCOIN.NET)

$LAZY

20/02/2022

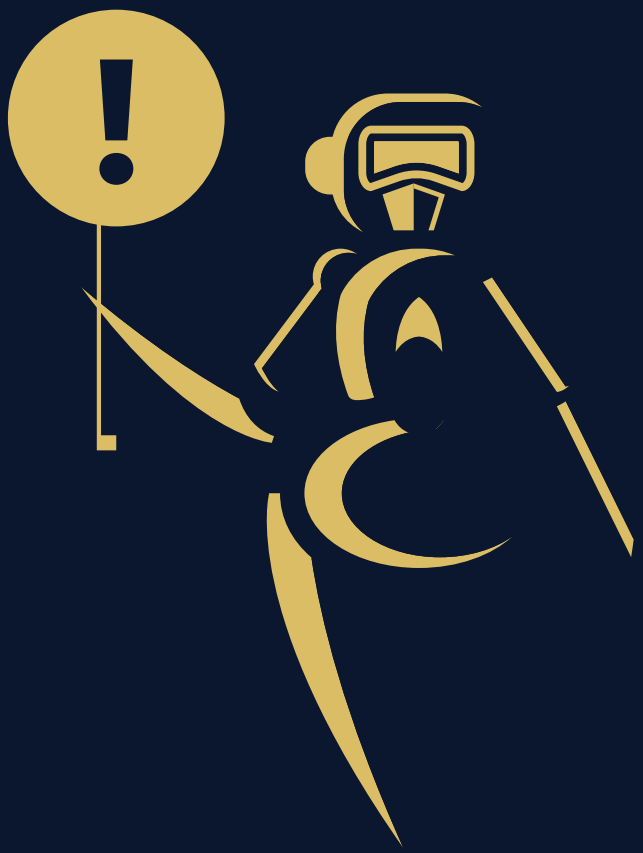
TABLE OF CONTENTS

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
Lazy Coin (LAZYCOIN.NET) (Customer) to conduct a Smart Contract Code Revie
and Security Analysis.

0x248F776ee3AC26B5aDC961a0CA94e7f062dec9FB

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 20/02/2022

WEBSITE DIAGNOSTIC

https://lazycoin.net/

0-49 50-89 90-100

Performance	Accessibility	Best Practices	SEO	Progressive Web App
78	95	90	93	NA

Metrics

First Contentful Paint
3.4 s

Time to interactive
7.3 s

Speed Index
11.3 s

Total Blocking Time
510 ms

Large Contentful Paint
5.9 s

Cumulative Layout Shift
0.053

WEBSITE IMPROVEMENTS

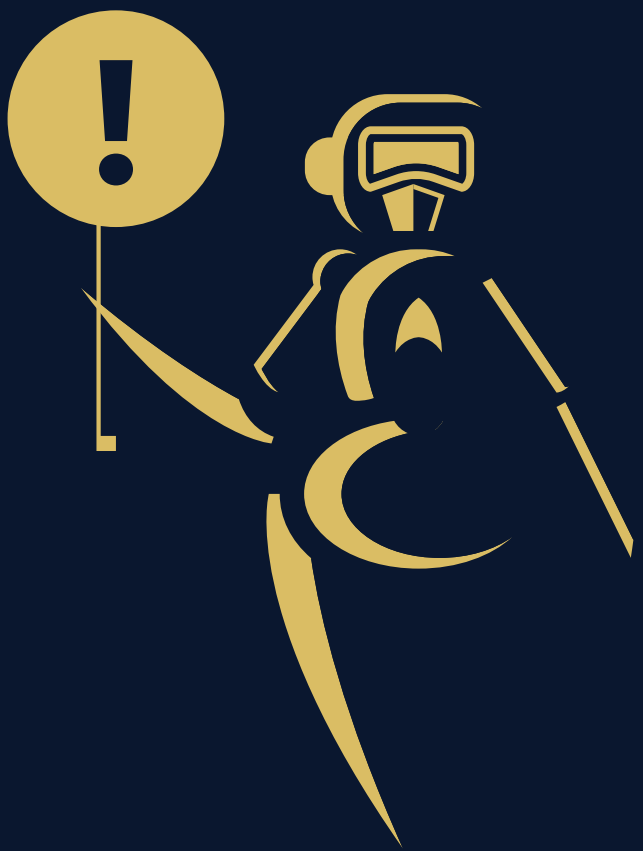
RReduce unused CSS

Reduce unused JavaScript

Links do not have a discernible name

Heading elements are not in a sequentially-descending order

Document does not have a meta description



AUDIT OVERVIEW

94

Security Score

92

Static Scan
Automatic scanning for common vulnerabilities

97

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/include wallet from fees

```
function excludeFromFee(address account) public onlyOwner() {
    _isExcludedFromFee[account] = true;
}

function includeInFee(address account) public onlyOwner() {
    _isExcludedFromFee[account] = false;
}
```

Contract owner can exclude/include wallet from rewards

```
function excludeFromReward(address account) external onlyOwner() {
    require(!_isExcluded[account], "Account is already excluded");
    if(_rOwned[account] > 0) {
      _tOwned[account] = tokenFromReflection(_rOwned[account]);
    }
    _isExcluded[account] = true;
    _excluded.push(account);
}

function includeInReward(address account) external onlyOwner() {
    require(_isExcluded[account], "Account is already included");
    for (uint256 i = 0; i < _excluded.length; i++) {
      if (_excluded[i] == account) {
        _excluded[i] = _excluded[_excluded.length - 1];
        _tOwned[account] = 0;
        _isExcluded[account] = false;
        _excluded.pop();
        break;
      }
    }
}
```

Contract owner can change buyback settings

```
function setBuyback(bool b) external onlyOwner() {
    _enableBuyback = b;
}

function setBuybackBNBThreshold(uint256 bnbAmount) external onlyOwner() {
    _buybackBNBThreshold = bnbAmount;
}

function setBuybackUpperLimit(uint256 buybackLimit) external onlyOwner() {
    _buybackUpperLimit = buybackLimit;
}

function setBuybackBNBPercentage(uint256 percentage) external onlyOwner() {
    _buybackBNBPercentage = percentage;
}
```

Contract owner can change swap settings

```
function setLiquidity(bool b) external onlyOwner() {
        _enableLiquidity = b;
}
```

Contract owner can change lottery settings

```
function setLottery(bool b) external onlyOwner() {
        _enableLottery = b;
}

function setLotteryChance(uint chance) external onlyOwner() {
        _lotteryChance = chance;
}

function setLotteryThreshold(uint256 threshold) external onlyOwner() {
        _lotteryThreshold = threshold;
}

function setLotteryMinimumSpend(uint256 minimumSpend) external onlyOwner() {
        _lotteryMinimumSpend = minimumSpend;
}
```

Contract owner can change _marketingAddress address

Current address:

_marketingAddress: 0x24ec43d7c90c7c21a12c7a0c18d29cc20d91e0d9

```
function setMarketingAddress(address marketingAddress) external onlyOwner() {
        _marketingAddress = payable(marketingAddress);
}
```

Contract owner can change max tx amount

```
function setMaxTxAmount(uint256 maxTxAmount) external onlyOwner() {
        require(maxTxAmount >= (_tTotal.mul(1).div(10000)).div(10**18), "amount must be greater than 0.01%
of the total supply");
        _maxTxAmount = maxTxAmount;
}
```

Contract owner can renounce ownership

```
function renounceOwnership() public virtual onlyOwner {
        _setOwner(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");
        _setOwner(newOwner);
}
```

Contract owner can change the fees up to 50%

```
function setTaxFeePercent(uint256 taxFee) external onlyOwner() {
        require(taxFee <= 28, "Total fee is over 28%");
        _taxFee = taxFee;
}

function setSellTaxFeePerecent(uint256 taxFee) external onlyOwner() {
        require(taxFee <= 28, "Total fee is over 28%");
        _sellTaxFee = taxFee;
}

function setWhaleSellTaxFeePerecent(uint256 taxFee) external onlyOwner() {
        require(taxFee <= 50, "Total fee is over 50%");
        _whaleSellTaxFee = taxFee;
}
```

Contract owner can change _whaleSellThreshold, _tokenSwapThreshold and _whaleSellTimer

```
function setTokenSwapThreshold(uint256 tokenSwapThreshold) external onlyOwner() {
        _tokenSwapThreshold = tokenSwapThreshold;
}

function setWhaleSellThreshold(uint256 amount) external onlyOwner() {
        _whaleSellThreshold = amount;
}

function setWhaleSellTimer(uint time) external onlyOwner() {
        _whaleSellTimer = time;
}
```

Contract owner can withdraw BNB or non-Lazy Coin tokens from the contract

```
function withdrawBNB(uint256 amount) public onlyOwner() {
        if(amount == 0) payable(owner()).transfer(address(this).balance);
        else payable(owner()).transfer(amount);
}

function withdrawForeignToken(address token) public onlyOwner() {
        require(address(this) != address(token), "Cannot withdraw native token");
        IERC20(address(token)).transfer(msg.sender, IERC20(token).balanceOf(address(this)));
}
```

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:	7%
Sell fees:	14-21%
Max TX:	N/A
Max Sell:	N/A

Honeypot Risk

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

Rug Pull Risk

Liquidity:	N/A
Holders:	Clean

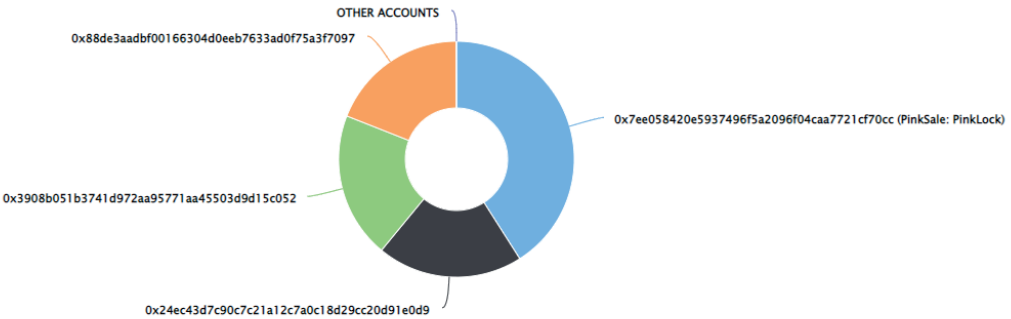
LAZY COIN TOKEN DISTRIBUTION & TOP 10 TOKEN HOLDERS

The top 10 holders collectively own 100.00%
(1,000,000,000,000,000.00 Tokens) of Lazy Coin (LAZYCOIN.NET)

Token Total Supply: 1,000,000,000,000,000.00 Token | Total Token Holders: 4

Lazy Coin (LAZYCOIN.NET) Top 10 Token Holders

Source: BscScan.com



(A total of 1,000,000,000,000,000.00 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	PinkSale: PinkLock	409,805,381,976,275.7	40.9805%
2	0x24ec43d7c90c7c21a12c7a0c18d29cc20d91e0d9	200,000,000,000,000	20.0000%
3	0x3908b051b3741d972aa95771aa45503d9d15c052	200,000,000,000,000	20.0000%
4	0x88de3aadbf00166304d0eeb7633ad0f75a3f7097	190,194,618,023,724.3	19.0195%

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.