freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

ikolF

$IKOLF

31/07/2022

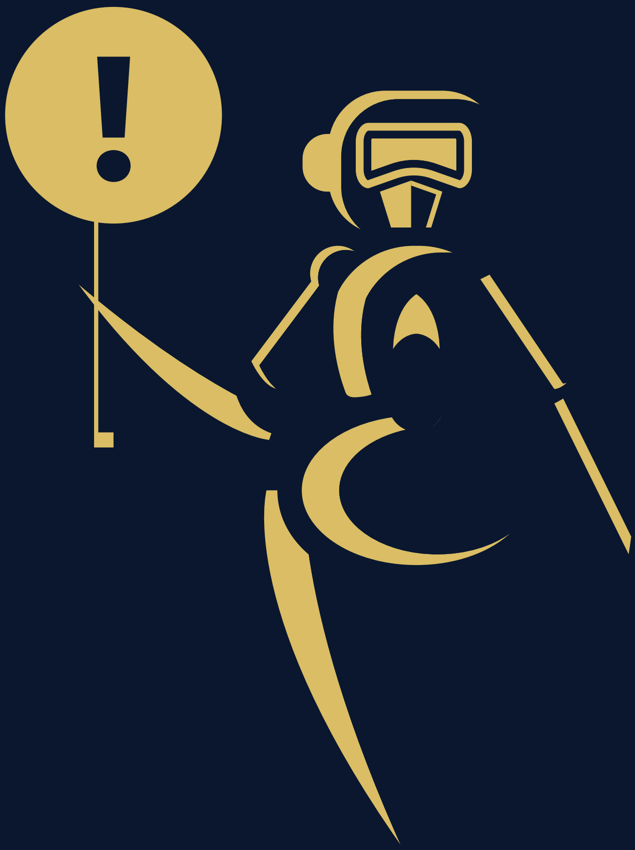
TABLE OF CONTENTS

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

0x49A516BD4406b2D4074C738a58De6DB397D0ABC9

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 31/07/2022

AUDIT OVERVIEW

92

Security Score

88

Static Scan
Automatic scanning for common vulnerabilities

98

ERC Scan
Automatic checks for ERC's conformance

0	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 address from tax

```
function setIsFeeExempt(address holder, bool exempt) external authorized {
        isFeeExempt[holder] = exempt;
}
```

Contract owner can exclude/include address from tx limitations

```
function setIsTxLimitExempt(address holder, bool exempt) external authorized {
        isTxLimitExempt[holder] = exempt;
}
```

Contract owner can exclude/include address from wallet limitations

```
function setIsWalletLimitExempt(address holder, bool exempt) external authorized {
        isWalletLimitExempt[holder] = exempt;
}
```

Contract owner can change max wallet amount

```
function setMaxWalletPercent(uint256 maxWallPercent) external onlyOwner() {
        _maxWalletToken = (_totalSupply * maxWallPercent ) / 100;
}
```

Contract owner can change max tx amount

```
function setTxLimit(uint256 amount) external authorized {
        require(amount >= (_totalSupply/100)*1, "Cannot set below 1%");
        _maxTxAmount = amount;
}
```

Contract owner can change swap settings

```
function setSwapBackSettings(bool _enabled, uint256 _amount) external authorized {
        swapEnabled = _enabled;
        swapThreshold = _amount;
}
```

Contract owner can change fees up to 15%

```
function setFees(uint256 _liquidityFee, uint256 _marketingFee) external authorized {
        require(totalFee < 15);
        liquidityFee = _liquidityFee;
        marketingFee = _marketingFee;
        totalFee = _liquidityFee.add(_marketingFee);
}
```

Contract owner can change autoLiquidityReceiver and marketingFeeReceiver addresses

```
function setFeeReceivers(address _autoLiquidityReceiver, address _marketingFeeReceiver) external autho-
rized {
        autoLiquidityReceiver = _autoLiquidityReceiver;
        marketingFeeReceiver = _marketingFeeReceiver;
}
```

Contract owner can transfer ownership

```
function transferOwnership(address payable adr) public onlyOwner {
        owner = adr;
        authorizations[adr] = true;
        emit OwnershipTransferred(adr);
}
```

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 issue during the first review.

TOKEN DETAILS

Details

Buy fees:	10%
Sell fees:	10%
Max TX:	1,000,000,000
Max Sell:	N/A

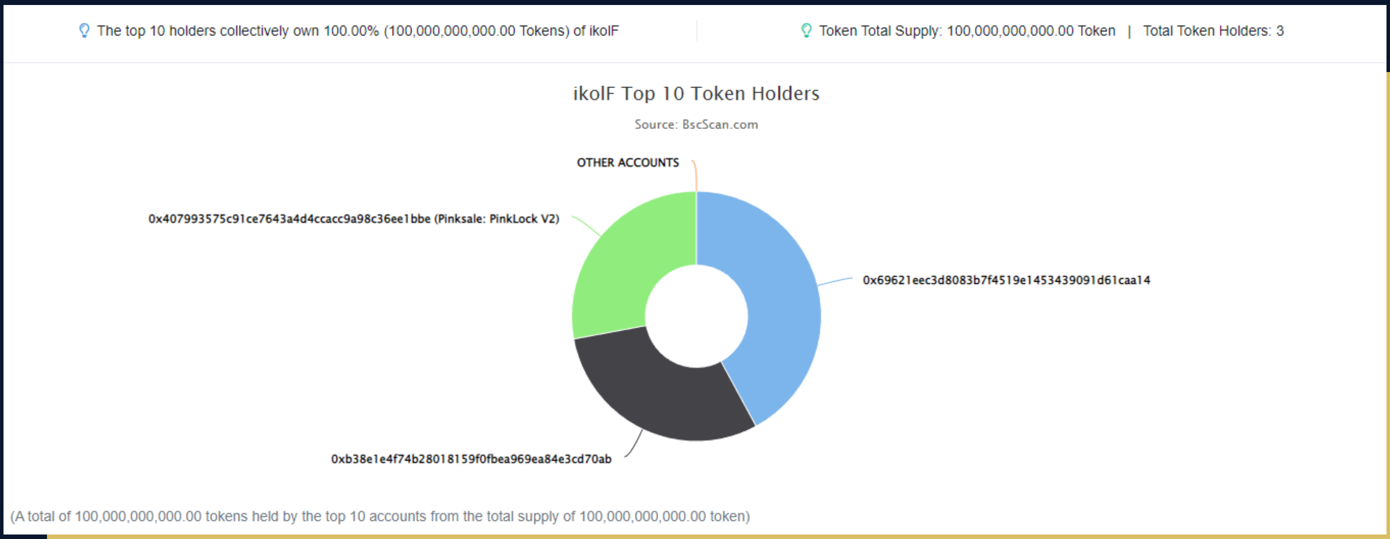
Honeypot Risk

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

Rug Pull Risk

Liquidity:	N/A
Holders:	Clean

IKOLF TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

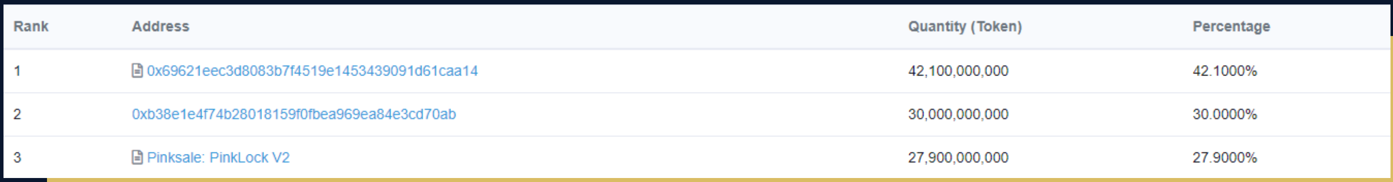


The top 10 holders collectively own 100.00% (100,000,000,000.00 Tokens) of ikolF | Token Total Supply: 100,000,000,000.00 Token | Total Token Holders: 3

ikolF Top 10 Token Holders

Source: BscScan.com

(A total of 100,000,000,000.00 tokens held by the top 10 accounts from the total supply of 100,000,000,000.00 token)



Rank	Address	Quantity (Token)	Percentage
1	0x69621eec3d8083b7f4519e1453439091d61caa14	42,100,000,000	42.1000%
2	0xb38e1e4f74b28018159f0fbea969ea84e3cd70ab	30,000,000,000	30.0000%
3	Pinksale: PinkLock V2	27,900,000,000	27.9000%

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.