



SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT



Easter Floki
\$FLOE

26/03/2023



TOKEN OVERVIEW

Fees

- Buy fees: 9%
- Sell fees: 9%

Fees privileges

- Can change buy fees up to 20% and sell fees up to 20%

Ownership

- Owned

Minting

- No mint function

Max Tx Amount / Max Wallet Amount

- Can change max tx amount (with threshold)

Blacklist

- Blacklist function not detected

Other privileges

- Can exclude / include from fees
-

TABLE OF CONTENTS

1

DISCLAIMER

2

INTRODUCTION

3

WEBSITE + SOCIALS

4-5

AUDIT OVERVIEW

6-9

OWNER PRIVILEGES

10

CONCLUSION AND ANALYSIS

11

TOKEN DETAILS

12

FLOE TOKEN ANALYTICS &
TOP 10 TOKEN HOLDERS

13

TECHNICAL DISCLAIMER



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 **Easter Floki** (Customer) to conduct a Smart Contract Code Review and Security Analysis.

0x3F69570634Bac492Ad78CE7dC11Bc3b3F298c97f

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 **26/03/2023**



WEBSITE DIAGNOSTIC

<https://easter-floki.com/>



0-49



50-89



90-100



Performance



Accessibility



Best
Practices



SEO



Progressive
Web App

Socials



Twitter

<https://twitter.com/easterflokii>



Telegram

https://t.me/EasterFloki_BSC

AUDIT OVERVIEW



Security Score



Static Scan

Automatic scanning for common vulnerabilities



ERC Scan

Automatic checks for ERC's conformance



High



Medium



Low



Optimizations



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 wallet from tx limitations**

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

- **Contract owner can exclude wallet from all limits (fees, tx limitations)**

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

- **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;
}
```

- The liquidity of the contract automatically gets credited into the owner's wallet whenever the 'addLiquidity' function is called inside the contract.

Note that it cannot be called manually but it will be done automatically every time the swap and liquify function is called. Moreover, even after the renouncement of the ownership, this liquidity will still be credited to the owner's wallet.

```
function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {  
    // approve token transfer to cover all possible scenarios  
    _approve(address(this), address(uniswapV2Router), tokenAmount);  
  
    // add the liquidity  
    uniswapV2Router.addLiquidityETH{value: ethAmount}(  
        address(this),  
        tokenAmount,  
        0, // slippage is unavoidable  
        0, // slippage is unavoidable  
        owner(),  
        block.timestamp  
    );  
}
```

- Contract owner can change buy fees up to 20% and sell fees up to 20%

```
function updateBuyFees(uint256 _marketingFee, uint256 _liquidityFee, uint256 _buyBackFee) external  
onlyOwner {  
    buyMarketingFee = _marketingFee;  
    buyLiquidityFee = _liquidityFee;  
    buyBuyBackFee = _buyBackFee;  
    buyTotalFees = buyMarketingFee + buyLiquidityFee + buyBuyBackFee;  
    require(buyTotalFees <= 20, "Must keep fees at 20% or less");  
}  
  
function updateSellFees(uint256 _marketingFee, uint256 _liquidityFee, uint256 _buyBackFee) external  
onlyOwner {  
    sellMarketingFee = _marketingFee;  
    sellLiquidityFee = _liquidityFee;  
    sellBuyBackFee = _buyBackFee;  
    sellTotalFees = sellMarketingFee + sellLiquidityFee + sellBuyBackFee;  
    require(sellTotalFees <= 20, "Must keep fees at 20% or less");  
}
```

- Contract owner can change max tx limitations (with threshold)

```
function updateMaxAmount(uint256 newNum) external onlyOwner {  
    require(newNum >= (totalSupply() * 5 / 1000) / 1e18, "Cannot set maxTransactionAmount lower than  
0.5%");  
    maxTransactionAmount = newNum * (10**18);  
}
```

● Contract owner can change **marketingWallet** and **buyBackWallet** addresses

Current values:

marketingWallet : 0x2f261248bcb4c750998632fcba48bc806a5b109c

buyBackWallet : 0x8ac61237ae9c789de4fb1e85e346ae950b508923

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

function updateBuyBackWallet(address newWallet) external onlyOwner {
    emit buyBackWalletUpdated(newWallet, buyBackWallet);
    buyBackWallet = newWallet;
}
```

● Contract owner can withdraw tokens from smart contract (FLOE excluded)

```
function withdrawStuckBNB() external onlyOwner{
    require(address(this).balance > 0, "Can't withdraw negative or zero");
    payable(owner()).transfer(address(this).balance);
}

function removeStuckToken(address _address) external onlyOwner {
    require(_address != address(this), "Can't withdraw tokens destined for liquidity");
    require(IERC20(_address).balanceOf(address(this)) > 0, "Can't withdraw 0");

    IERC20(_address).transfer(owner(), IERC20(_address).balanceOf(address(this)));
}
```

● 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:	9%
Sell fees:	9%
Max TX:	10,000,000,000
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:	100% tokens unlocked



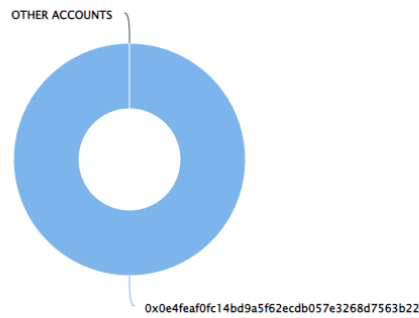
FLOE TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

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

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

Easter Floki Top 10 Token Holders

Source: BscScan.com



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

Rank	Address	Quantity (Token)	Percentage
1	0x0e4feaf0fc14bd9a5f62ecdb057e3268d7563b22	1,000,000,000,000	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.

