freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Valentine Floki

$FLOV

14/01/2023

TOKEN OVERVIEW

Fees

• Buy fees:	3%
• Sell fees:	3%

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 or max wallet amount (with threshold)

Blacklist

- Blacklist function not detected

Other privileges

- Can exclude / include from fees

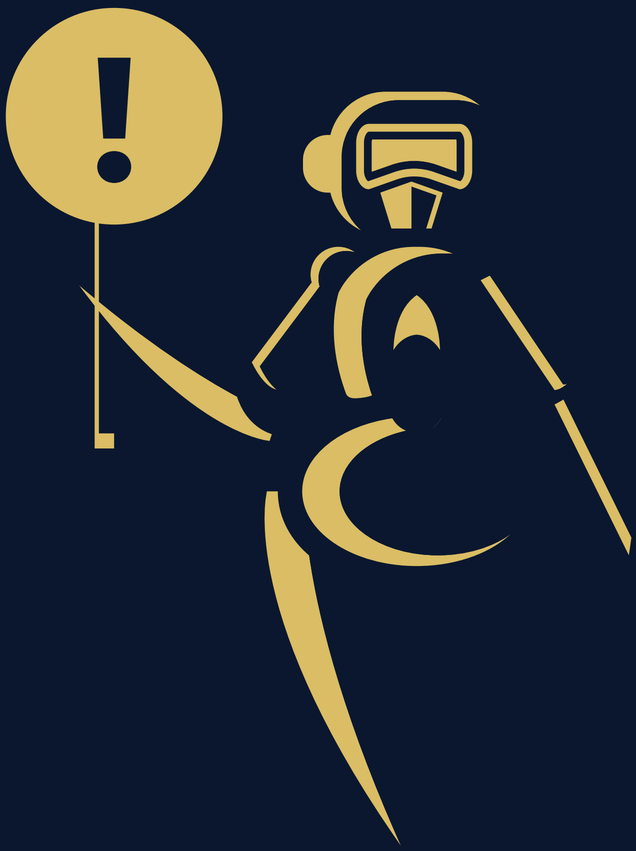
TABLE OF CONTENTS

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

0xE94760c58e2175E369722372Beea68395c645dc9

Network: Ethereum (ETH)

This report presents the findings of the security assessment of Customer's smart contract and its code review conducted on 14/01/2023

AUDIT OVERVIEW

92

Security Score

89

Static Scan

Automatic scanning for common vulnerabilities

92

ERC Scan

Automatic checks for ERC's conformance

0	High
2	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 from tax

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

🟡 Contract owner can exclude/include wallet from tx limitations

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

🟢 Contract owner can enable/disable wallet limitations

```
function enableMaxWalletLimit() external onlyOwner {
        require(maxWalletLimit == false, "Already max wallet limit is enabled");
        maxWalletLimit = true;
}

function disableMaxWalletLimit() external onlyOwner {
        require(maxWalletLimit == true, "Already max wallet limit is disabled");
        maxWalletLimit = false;
}
```

🟢 Contract owner can change max tx amount limitations (with threshold)

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

🟢 Contract owner can change max wallet amount limitations (with threshold)

```
function setMaxWallet(uint256 newAmount) external onlyOwner returns (bool) {
        maxWallet = newAmount *(10**18);
        require(maxWallet >= totalSupply() * 5 / 1000, "Swap amount cannot be lower than 0.5% total supply.");
        require(maxWallet <= totalSupply() * 2 / 100, "Swap amount cannot be higher than 2% total supply.");
        return true;
}
```

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

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

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

Contract owner can change swap settings

```
function updateSwapEnabled(bool enabled) external onlyOwner(){
      swapEnabled = enabled;
}

function updateSwapTokensAtAmount(uint256 newAmount) external onlyOwner returns (bool){
      swapTokensAtAmount = newAmount * (10**18);
      require(swapTokensAtAmount >= totalSupply() * 1 / 100000, "Swap amount cannot be lower than 0.001%
total supply.");
      require(swapTokensAtAmount <= totalSupply() * 5 / 1000, "Swap amount cannot be higher than 0.5%
total supply.");
      return true;
}
```

Contract owner can change marketingWallet address

Default value:

marketingWallet : 0xc53BE9b3356775EF96503B39c53B388D5BC290D1

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

Contract owner can withdraw stuck tokens from smart contract

```
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)));
}
```

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

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:	3%
Sell fees:	3%
Max TX:	1,000,000,000,000
Max Sell:	N/A

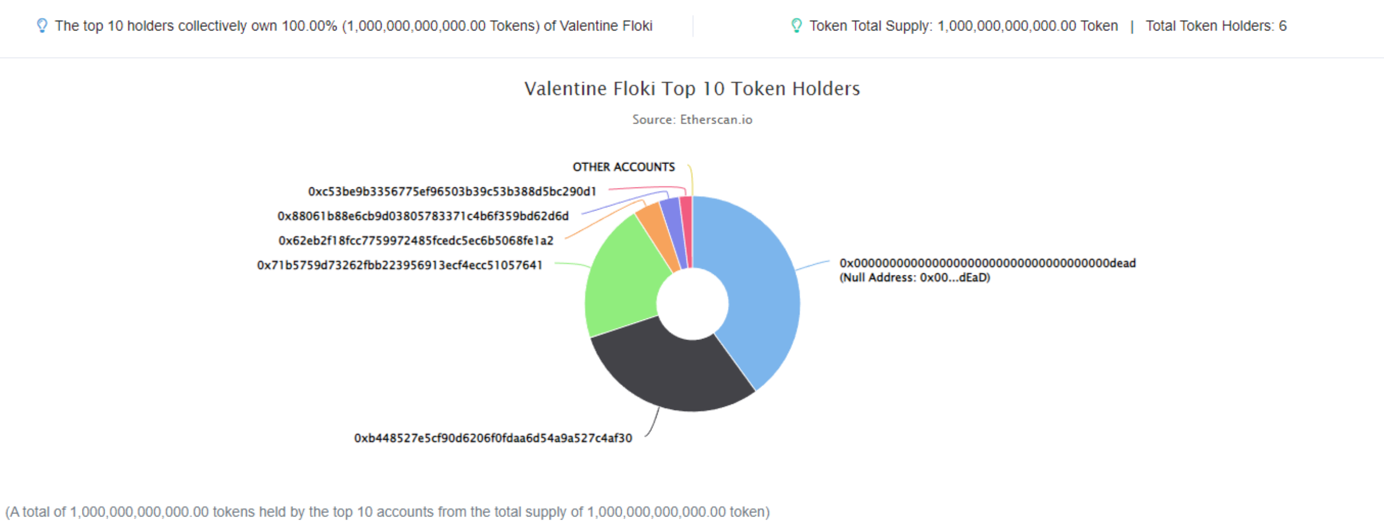
Honeypot Risk

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

VALENTINE FLOKI TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS



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

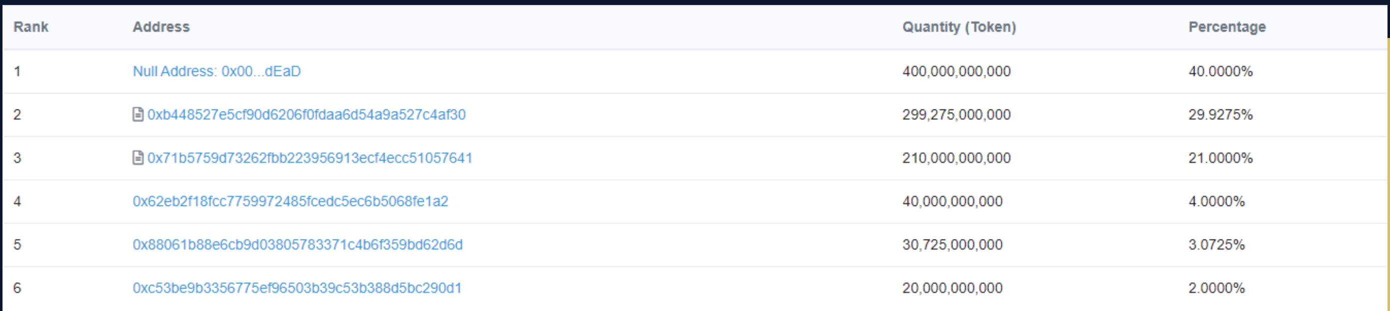
Token Total Supply: 1,000,000,000,000.00 Token | Total Token Holders: 6

Valentine Floki Top 10 Token Holders

Source: Etherscan.io

OTHER ACCOUNTS
0xc53be9b3356775ef96503b39c53b388d5bc290d1
0x88061b88e6cb9d03805783371c4b6f359bd62d6d
0x62eb2f18fcc7759972485fcedc5ec6b5068fe1a2
0x71b5759d73262fbb223956913ecf4ecc51057641
0x000000000000000000000000000000000000dead
(Null Address: 0x00...dEaD)
0xb448527e5cf90d6206f0fdaa6d54a9a527c4af30

(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	Null Address: 0x00...dEaD	400,000,000,000	40.0000%
2	0xb448527e5cf90d6206f0fdaa6d54a9a527c4af30	299,275,000,000	29.9275%
3	0x71b5759d73262fbb223956913ecf4ecc51057641	210,000,000,000	21.0000%
4	0x62eb2f18fcc7759972485fcedc5ec6b5068fe1a2	40,000,000,000	4.0000%
5	0x88061b88e6cb9d03805783371c4b6f359bd62d6d	30,725,000,000	3.0725%
6	0xc53be9b3356775ef96503b39c53b388d5bc290d1	20,000,000,000	2.0000%

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.