freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Virtualfields

$Fields

26/01/2022

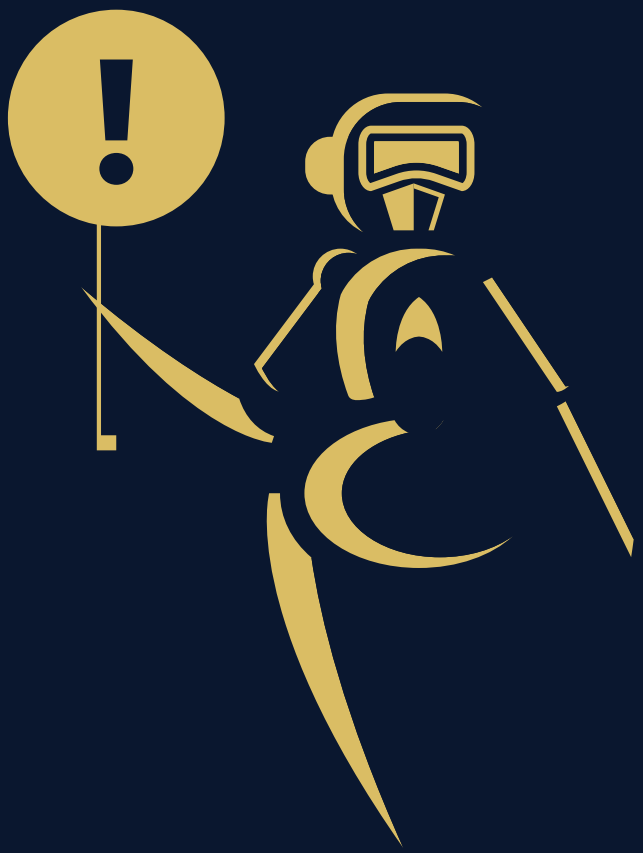
TABLE OF CONTENTS

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

0x99Ee534A77e213DcF9EaDbA34843faeFDf04e060

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 26/01/2022

WEBSITE DIAGNOSTIC

https://virtualfields.net/

0-49 50-89 90-100

Performance	Accessibility	Best Practices	SEO	Progressive Web App
86	95	91	90	NA

Metrics

First Contentful Paint
2.8 s

Time to interactive
4.5 s

Speed Index
4.2 s

Total Blocking Time
160 ms

Large Contentful Paint
2.9 s

Cumulative Layout Shift
0.006

WEBSITE IMPROVEMENTS

Reduce initial server response time

Reduce unused JavaScript

Reduce unused CSS

Image elements do not have explicit width and height

Ensure text remains visible during webfont load

Background and foreground colors do not have a sufficient contrast ratio

Heading elements are not in a sequentially-descending order

Links do not have a discernible name

AUDIT OVERVIEW

96

Security Score

96

Static Scan
Automatic scanning for common vulnerabilities

96

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.

Owner must be a contract with transparent rules for using setBots function

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 reward

```
function excludeFromReward(address account) public 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 excluded");
        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 swap settings

```
function setSwapAndLiquifyEnabled(bool _enabled) public onlyOwner {
        swapAndLiquifyEnabled = _enabled;
        emit SwapAndLiquifyEnabledUpdated(_enabled);
}
```

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;
}
```

Contract owner can change the buy and sell fees

```
function setBuyFees(uint256 taxFee, uint256 liquidityFee, uint256 devFee, uint256 charityFee)
  external onlyOwner() {
    setAllFees(taxFee, liquidityFee, devFee, charityFee);
    _previousTaxFee = taxFee;
    _previousLiquidityFee = liquidityFee;
    _previousCharityFee = charityFee;
    _previousDevFee = devFee;
}

function setAllFees(uint256 taxFee, uint256 liquidityFee, uint256 devFee, uint256 charityFee) private {
    _taxFee = taxFee;
    _liquidityFee = liquidityFee;
    _devFee = devFee;
    _charityFee = charityFee;
}
.
.
.
function setSaleFees(uint256 taxFee, uint256 liquidityFee, uint256 devFee, uint256 charityFee)
  external onlyOwner() {
    _saleTaxFee = taxFee;
    _saleLiquidityFee = liquidityFee;
    _saleDevFee = devFee;
    _saleCharityFee = charityFee;
}
```

Contract owner can change max tx amount and max tx wallet

```
function setMaxTxAmount(uint256 maxTxAmount) external onlyOwner() {
    require(maxTxAmount > 0, "transaction amount must be greater than zero");
    _maxTxAmount = maxTxAmount * (10**18);
}
.
.
.
function setMaxWalletTokend(uint256 _maxToken) external onlyOwner {
            maxWalletToken = _maxToken * (10**18);
}
```

Contract owner can change minimum tokens before swap

```
function setMinimumTokensBeforeSwap(uint256 newAmt) external onlyOwner() {
    minimumTokensBeforeSwap = newAmt * (10**18);
}
```

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:	15%
Sell fees:	15%
Max TX:	5,000,000,000
Max Sell:	N/A

Honeypot Risk

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

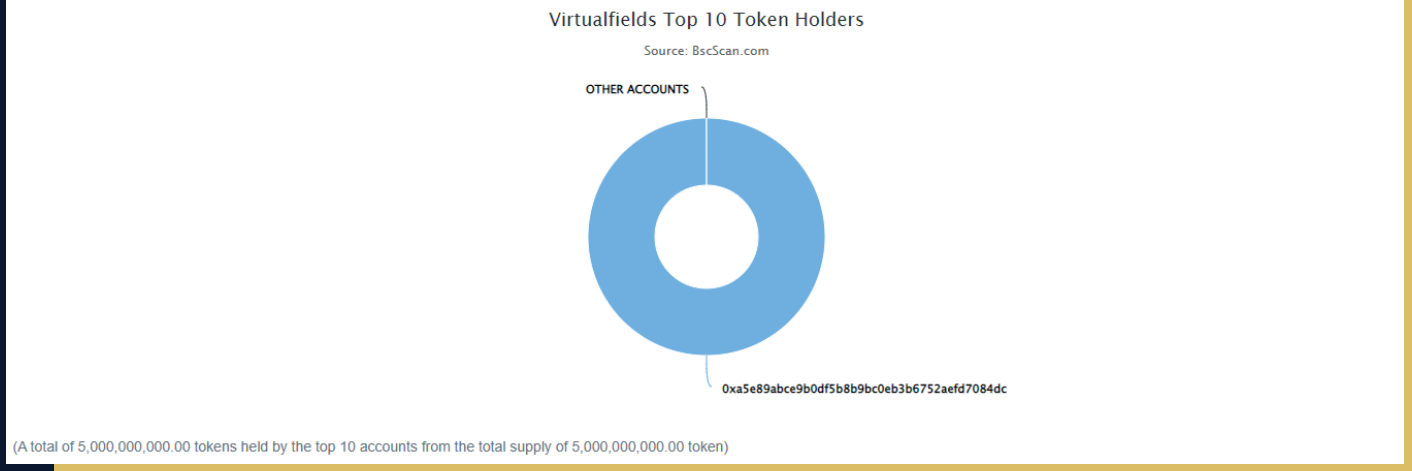
Rug Pull Risk

Liquidity:	N/A
Holders:	Clean

VIRTUALFIELDS TOKEN DISTRIBUTION & TOP 10 TOKEN HOLDERS

The top 10 holders collectively own 100.00% (5,000,000,000.00 Tokens) of Virtualfields

Token Total Supply: 5,000,000,000.00 Token | Total Token Holders: 1



(A total of 5,000,000,000.00 tokens held by the top 10 accounts from the total supply of 5,000,000,000.00 token)

Rank	Address	Quantity (Token)	Percentage
1	0xa5e89abce9b0df5b8b9bc0eb3b6752aefd7084dc	5,000,000,000	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.