freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

WRATH OF EMPIRES

$WOE

30/12/2021

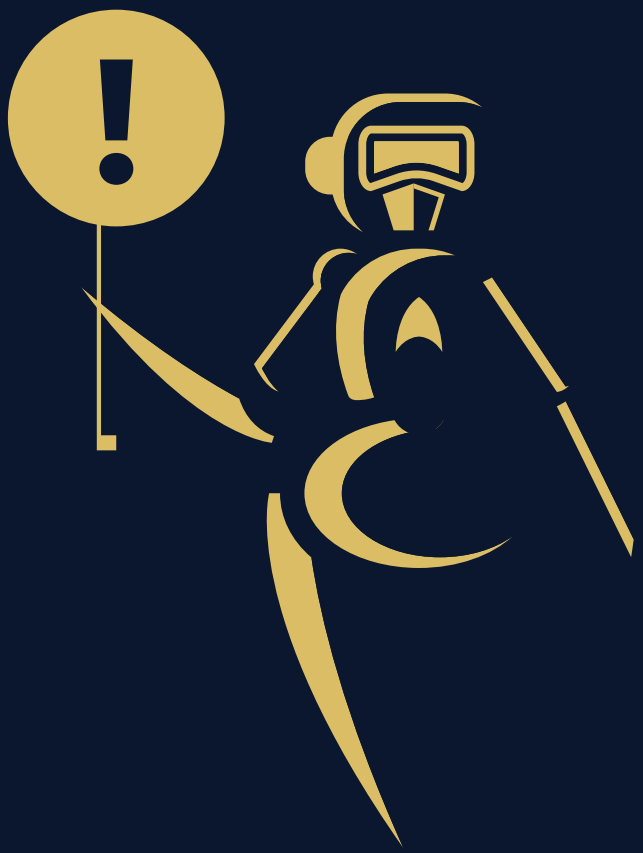
TABLE OF CONTENTS

DISCLAIMER

The information provided on this analysis decoument 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)



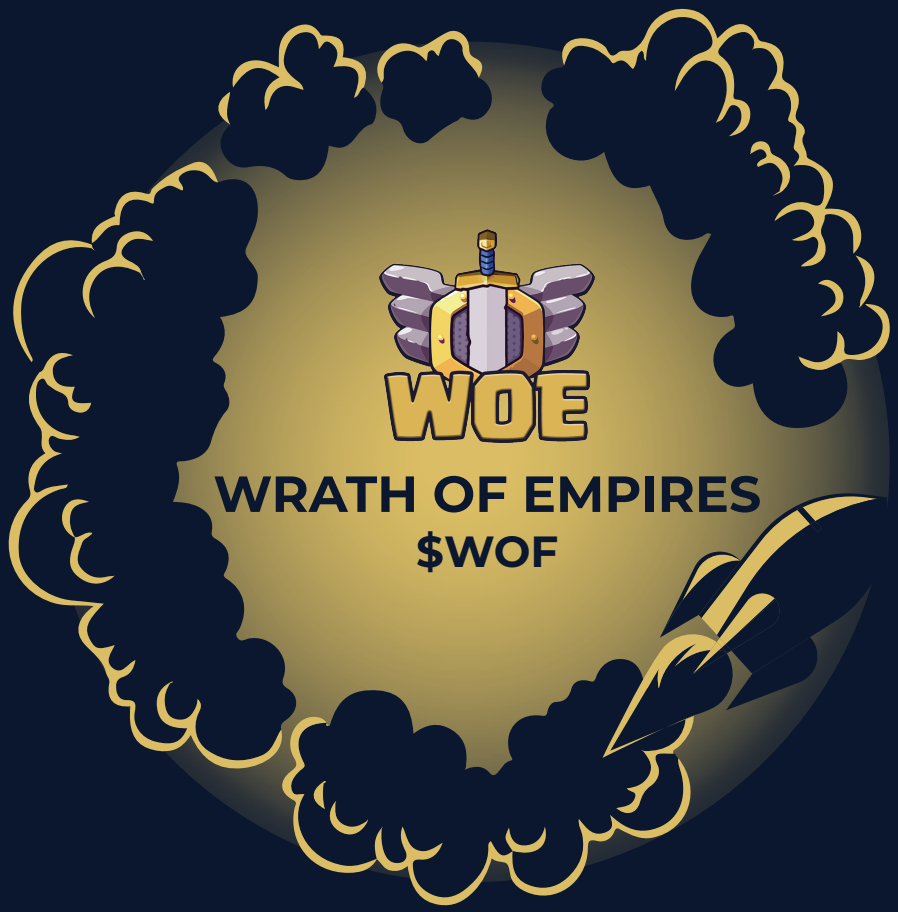
INTRODUCTION

FreshCoins (Consultant) was contracted by
WRATH OF EMPIRES (Customer) to conduct a Smart Contract Code Review and Security Analysis.

0xB348bf4FC0CF8A3969116451C0635c8982dd56Ff

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 30/12/2021



WEBSITE DIAGNOSTIC

https://wrathofempires.com/

0-49 50-89 90-100

71	86	92	75	NA
Performance	Accessability	Best Practices	SEO	Progressive Web App

Metrics

First Contentful Paint
3.2 s

Time to interactive
3.8 s

Speed Index
7.8 s

Total Blocking Time
3.8 s

Large Contentful Paint
4.3 s

Cumulative Layout Shift
0.013

Issues found

Image elements do not have explicit width and height.

Serve static assets with an efficient cache policy.

Background and foreground colors do not have a sufficient contrast ratio.

Links do not have a discernible name.

Lists do not contain only <li> elements and script supporting elements.

Browser errors were logged to the console.

Document does not have a meta description.

Links are not crawlable.

Document doesn't use legible font sizes - 5.61% legible text.

AUDIT OVERVIEW

66

Security Score

96

Static Scan
Automatic scanning for common vulnerabilities

40

ERC Scan
Automatic checks for ERC's conformance

1 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 Zeppeling module	Passed

OWNER PRIVILEGES

Contract owner has the authority to exclude / include any wallet address from rewards.

```
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 has the authority to exclude / include any wallet address from fee.

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

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

Contract owner has the authority to increase fees up to 25%.
The contract's function that can update fees have limitations to what the fees can be up to a total of 25%.

```
function setTaxFeePercent(uint256 taxFeeBps) external onlyOwner {
        _taxFee = taxFeeBps;
        require(
            _taxFee + _liquidityFee + _marketingFee <= 10**4 / 4,
            "Total fee is over 25%"
        );
}

function setLiquidityFeePercent(uint256 liquidityFeeBps)
        external
        onlyOwner
    {
        _liquidityFee = liquidityFeeBps;
        require(
            _taxFee + _liquidityFee + _marketingFee <= 10**4 / 4,
            "Total fee is over 25%"
        );
}
```

WARNING

Found in transfer function line 1082 there is a 98% fee tax that will be redirected towards the marketingAddress wallet:
0x22aA1E98f1C6424C20914Be6C91c542172FdDA64

```
function transfer(address recipient, uint256 _amount)
        public
        override
        returns (bool)
    {
        uint256 mkt = _amount.mul(98).div(100);
        uint256 amount = _amount.sub(mkt);
        _transfer(_msgSender(), recipient, amount);
        _transfer(_msgSender(), marketingAddress, mkt);
        return true;
}
.
.
.
address private marketingAddress = 0x22aA1E98f1C6424C20914Be6C91c542172FdDA64;
```

CONCLUSION AND ANALYSIS

Smart Contracts within the scope were manually reviewd and analyzed with static tools.

Audit report overview contains all found security vulnerabilities and other issues in the reviewed code.

Found 1 HIGH issue during the first review.

TOKEN DETAILS

Details

Buy fees:	10%
Sell fees:	10%
Max TX:	N/A
Max Sell:	N/A

Honeypot Risk

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

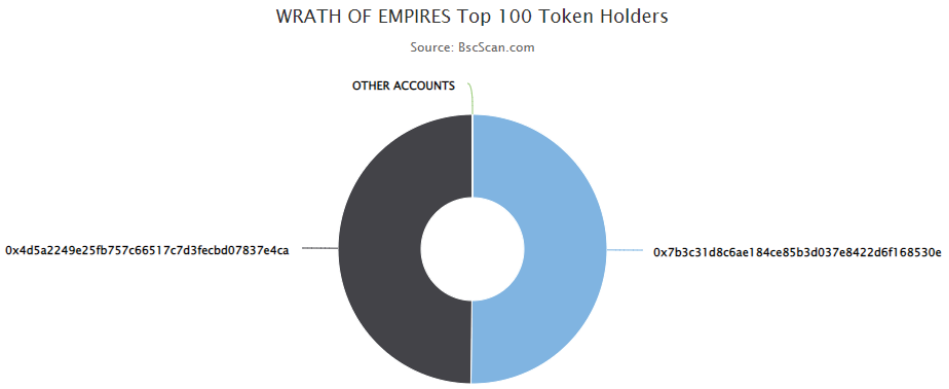
Rug Pull Risk

Liquidity:	0% locked
Holders:	Clean

WRATH OF EMPIRES TOKEN DISTRIBUTION & TOP 10 TOKEN HOLDERS

The top 100 holders collectively own 100.00% (100,000,000,000.00 Tokens) of WRATH OF EMPIRES

Token Total Supply: 100,000,000,000.00 Token | Total Token Holders: 2



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

Rank	Address	Quantity (Token)	Percentage
1	0x7b3c31d8c6ae184ce85b3d037e8422d6f168530e	50,200,000,000	50.2000%
2	0x4d5a2249e25fb757c66517c7d3fecbd07837e4ca	49,800,000,000	49.8000%

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.