freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

IMPERIAL GAMES

Imperial Games

$IMPG

19/02/2022

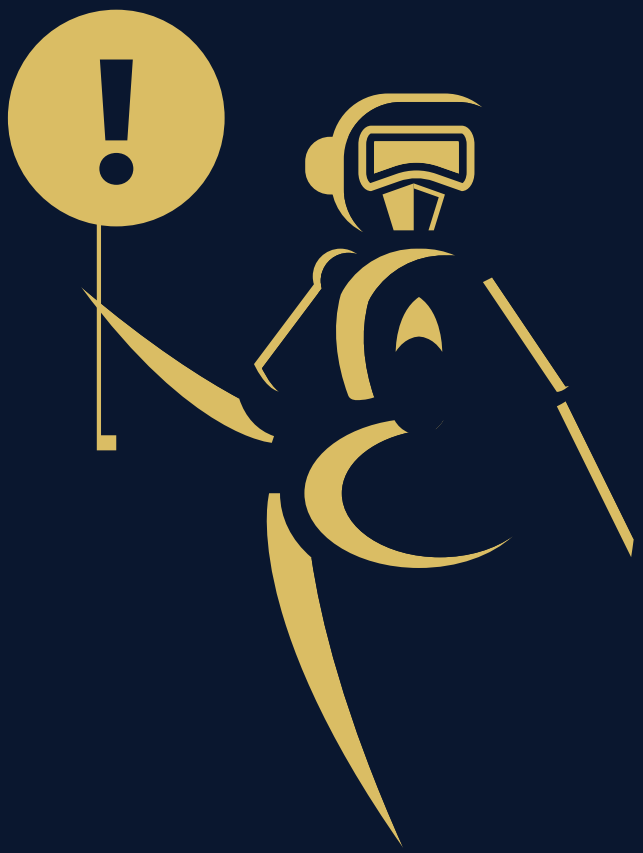
TABLE OF CONTENTS

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

0xA414C18869526cee2032F25842D0650B591B57d0

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 19/02/2022

WEBSITE DIAGNOSTIC

https://imperialgames.io/

0-49 50-89 90-100

94	96	92	98	NA
Performance	Accessibility	Best Practices	SEO	Progressive Web App

Metrics

First Contentful Paint
3.1 s

Time to interactive
4.1 s

Speed Index
5.1 s

Total Blocking Time
260 ms

Large Contentful Paint
2.9 s

Cumulative Layout Shift
0.012

AUDIT OVERVIEW

96

Security Score

94

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 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 rewards

```
function excludeFromReward(address account) public onlyOwner {
        // require(account != 0x7a250d5630B4cF539739dF2C5dAcb4c659F2488D, 'We can not exclude Uniswap router.');
        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 not 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 change liquidation process amount

```
function num2Add2LP(uint256 num2Add2Liquidity) external onlyOwner {
        numTokensSellToAddToLiquidity = num2Add2Liquidity;
}
```

Contract owner can change the fees up to 100%

```
function setFees(
      uint256 liquidityFee,
      uint256 Development,
      uint256 Marketing,
      uint256 burnFee,
      uint256 taxFee
   ) external onlyOwner {
      _liquidityFee = liquidityFee;
      _DevelopmentFee = Development;
      _MarketingFee = Marketing;
      _BurnFee = burnFee;
      _taxFee = taxFee;

      emit feesUpdated(liquidityFee, Development, Marketing, burnFee, taxFee);
}
```

Contract owner can change MarketingAdd and DevelopmentAdd addresses

Current values:

MarketingAdd : 0xca7405a58151c936433d2e05ccae9f17eab3efd3

DevelopmentAdd: 0x101c7973b922dc15681f7df477cacd3790b8cdda

```
function setMarketingAdd(address addr) external onlyOwner {
      MarketingAdd = addr;
}

function setDevelopmentAddress(address addr) external onlyOwner {
      DevelopmentAdd = addr;
}
```

Contract owner can change max tx amount

```
function setMaxTx(uint256 maxTx) external onlyOwner {
      _maxTxAmount = maxTx;
}
```

Contract owner can renounce ownership

```
function renounceOwnership() public virtual onlyOwner {
      _transferOwnership(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"
      );
      _transferOwnership(newOwner);
   }
```

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:	12%
Sell fees:	12%
Max TX:	100,000,000
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

IMPERIAL GAMES TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

The top 10 holders collectively own 100.00% (100,000,000,000.00 Tokens) of Imperial Games

Token Total Supply: 100,000,000,000.00 Token | Total Token Holders: 1

(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	0x289492da426d82a1d91d052c53468183a8645a24	100,000,000,000	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.