freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

For Teh Culture

$FTC

01/06/2023

TOKEN OVERVIEW

Fees

- Buy fees: 0%
- Sell fees: 0%

Fees privileges

- Can change buy fees up to 25% and sell fees up to 25%

Ownership

- Owned

Minting

- No mint function

Max Tx Amount / Max Wallet Amount

- Can change max tx amount and max wallet amount (with threshold)

Blacklist

- Blacklist function not detected

Other privileges

- Contract owner has to call launch() function to enable trade
- Can exclude / include from fees
- Can exclude / include from tx limitations

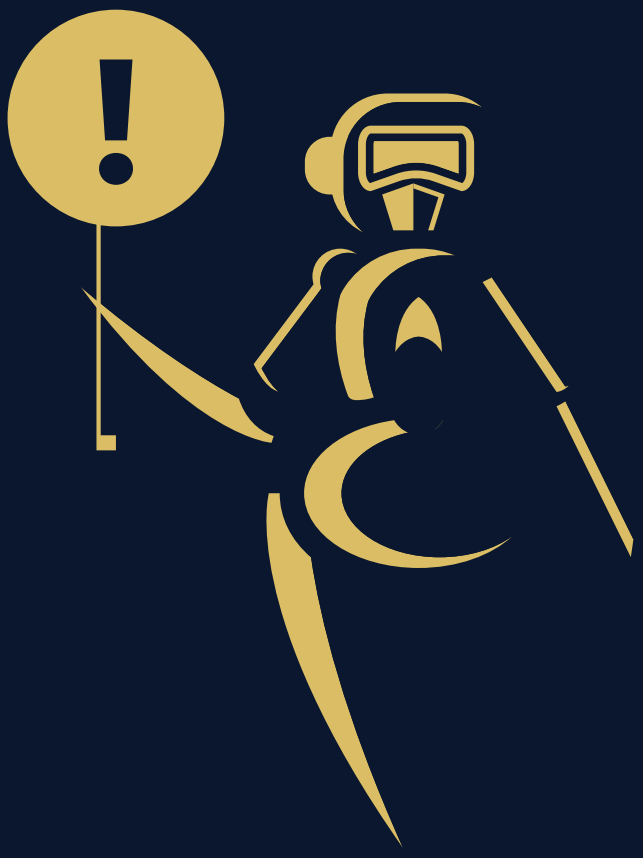
TABLE OF CONTENTS

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
For Teh Culture (Customer) to conduct a Smart Contract Code Review and Security Analysis.

0x4B7B224197965623FEc94bc6aA3F6aeeF9235Ba4

Network: Ethereum (ETH)

This report presents the findings of the security assessment of Customer's smart contract and its code review conducted on 01/06/2023

WEBSITE DIAGNOSTIC

N/A


0-49


50-89


90-100

Performance

Accessibility

Best Practices

SEO

Progressive Web App

Socials


Twitter

N/A


Telegram

N/A

AUDIT OVERVIEW

82

Security Score

98

Static Scan
Automatic scanning for common vulnerabilities

84

ERC Scan
Automatic checks for ERC's conformance

1 High

3 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	Low
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 has to call launch() function to enable trade

Please note that any wallet excluded from fees/taxes can still engage in trading even if trading is disabled.

```
function launch() public onlyOwner {
        require(launchedAt == 0, "Already launched boi");
        launchedAt = block.number;
        launchedAtTimestamp = block.timestamp;
        tradingActive = true;
        swapEnabled = true;
}

_transfer function line 915-928

if (limitsInEffect) {
            if (
                from != owner() &&
                to != owner() &&
                to != address(0) &&
                to != address(0xdead) &&
                !swapping
            ) {
                if (!tradingActive) {
                    require(
                        _isExcludedFromFees[from] || _isExcludedFromFees[to],
                        "Trading is not active."
                    );
                }
```

Contract owner can change buy fees up to 25% and sell fees up to 25%

```
function updateBuyFees(uint256 _marketingFeeOnBuy) external onlyOwner {
    buyMarketingFee = _marketingFeeOnBuy;
    require(buyMarketingFee <= 25, "fee must not be greater than 25%");
}

function updateSellFees(uint256 _marketingFeeOnSell) external onlyOwner {
    sellMarketingFee = _marketingFeeOnSell;
    require(sellMarketingFee <= 25, "fee must not be greater than 25%");
}
```

Contract owner can change marketingWallet address

Current value:

marketingWallet: 0xE35a9eB4aB4b210eeBD77C9B91a82556365f0952

```
function updateMarketingWallet(
    address newMarketingWallet
  ) external onlyOwner {
    require(
      marketingWallet != address(0),
      "_marketWallet address cannot be 0"
    );

    marketingWallet = newMarketingWallet;
    emit marketingWalletUpdated(newMarketingWallet, marketingWallet);
}
```

Contract owner can change max tx limitations and max wallet limitations (with threshold)

```
function updateMaxTxnAmount(uint256 newNum) external onlyOwner {
    require(
      newNum >= ((totalSupply() * 5) / 1000) / 1e18,
      "Cannot set max wallet amount lower than 0.5%"
    );
    require(
      newNum <= ((totalSupply() * 5) / 100) / 1e18,
      "Cannot set max wallet amount higher than 5%"
    );
    maxTransactionAmount = newNum * (10 ** 18);
}

function updateMaxWalletAmount(uint256 newNum) external onlyOwner {
    require(
      newNum >= ((totalSupply() * 5) / 1000) / 1e18,
      "Cannot set max wallet amount lower than 0.5%"
    );
    require(
      newNum <= ((totalSupply() * 5) / 100) / 1e18,
      "Cannot set max wallet amount higher than 5%"
    );
    maxWallet = newNum * (10 ** 18);
}
```

Contract owner can change swap settings

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

Contract owner has the authority to remove all limitations

```
function removeLimits() external onlyOwner returns (bool) {
        limitsInEffect = false;
        return true;
}

_transfer function line 915-928

if (limitsInEffect) {
            if (
               from != owner() &&
               to != owner() &&
               to != address(0) &&
               to != address(0xdead) &&
               !swapping
            ) {
               if (!tradingActive) {
                  require(
                     _isExcludedFromFees[from] || _isExcludedFromFees[to],
                     "Trading is not active."
                  );
               }
```

Contract owner has the ability to send tokens to multiple wallets using a single transaction

If the airdrop operation involves a large number of recipients, it may consume a significant amount of gas, potentially leading to out-of-gas errors

```
function airdrop(
        address[] calldata addresses,
        uint256[] calldata amounts
    ) external onlyOwner {
        require(
           addresses.length == amounts.length,
           "Array sizes must be equal"
        );
        uint256 i = 0;
        while (i < addresses.length) {
           uint256 _amount = amounts[i].mul(1e18);
           _transfer(msg.sender, addresses[i], _amount);
           i += 1;
        }
}
```

Contract owner can withdraw tokens from smart contract

Native tokens ($FTC) not excluded

```
function withdrawETH(uint256 _amount) external onlyOwner {
    require(address(this).balance >= _amount, "Invalid Amount");
    payable(msg.sender).transfer(_amount);
}

function withdrawToken(IERC20 _token, uint256 _amount) external onlyOwner {
    require(_token.balanceOf(address(this)) >= _amount, "Invalid Amount");
    _token.transfer(msg.sender, _amount);
}
```

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);
}

function _transferOwnership(address newOwner) internal virtual {
    address oldOwner = _owner;
    _owner = newOwner;
    emit OwnershipTransferred(oldOwner, newOwner);
}
```

Contract owner can renounce ownership

```
function renounceOwnership() public virtual onlyOwner {
    _transferOwnership(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 1 HIGH issues during the first review.

TOKEN DETAILS

Details

Buy fees:	0%
Sell fees:	0%
Max TX:	10,000,000
Max Sell:	30,000,000

Honeypot Risk

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

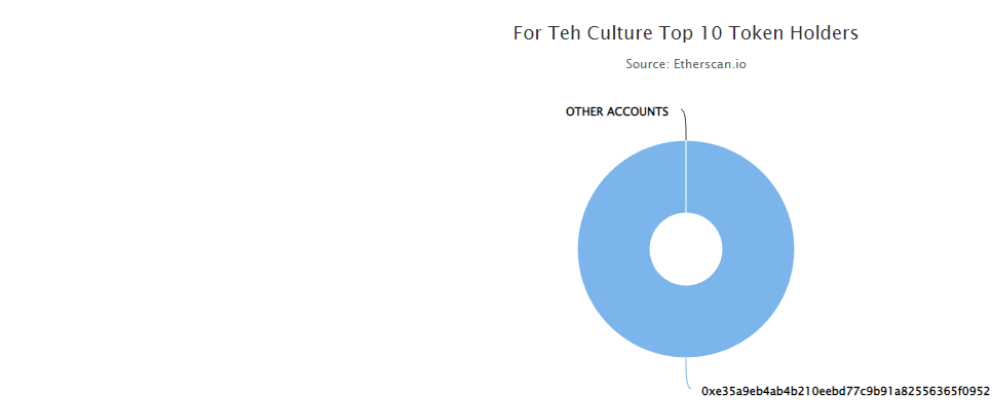
Rug Pull Risk

Liquidity:	N/A
Holders:	100% unlocked tokens

FTC TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

The top 10 holders collectively own 100.00% (1,000,000,000.00 Tokens) of For Teh Culture

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



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

Rank	Address	Quantity (Token)	Percentage
1	0xE35a9e...365f0952	1,000,000,000	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.