freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Meta Meerkat

$METAKAT

09/02/2022

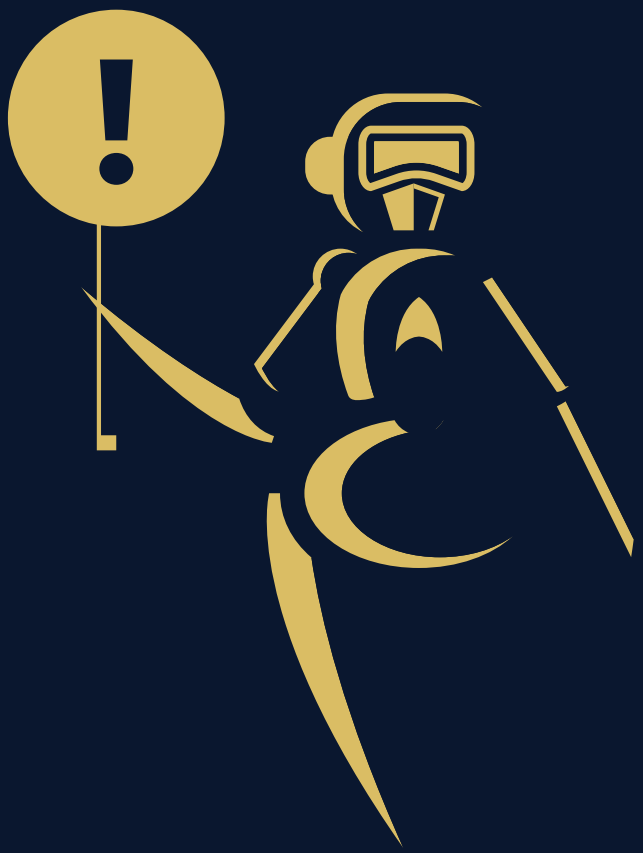
TABLE OF CONTENTS

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

0xAE8e248f75C1D0e75476A3820B819dFB86Ea16D8

Network: Binance Smart Chain (BNB)

This report presents the findings of the security assessment of Customer's smart contract and its code review conducted on 09/02/2022

WEBSITE DIAGNOSTIC

https://metakat.finance/

0-49 50-89 90-100

Performance	Accessibility	Best Practices	SEO	Progressive Web App
99	97	100	86	NA

Metrics

First Contentful Paint
1.4 s

Time to interactive
1.4 s

Speed Index
1.4 s

Total Blocking Time
0 ms

Large Contentful Paint
1.6 s

Cumulative Layout Shift
0

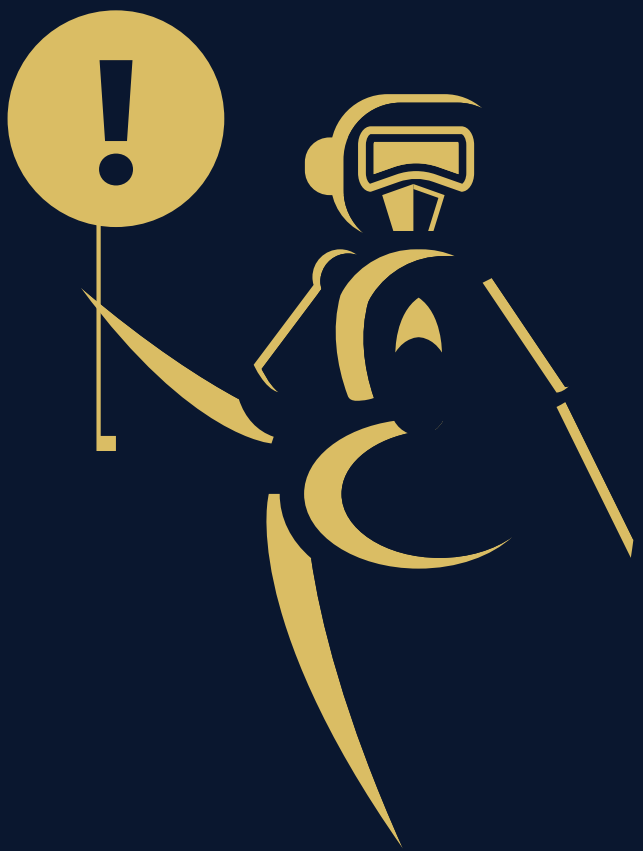
WEBSITE IMPROVEMENTS

Reduce unused CSS

Background and foreground colors do not have a sufficient contrast ratio

Document does not have a meta description

Links are not crawlable



AUDIT OVERVIEW

98

Security Score

98

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 exclude an address from transactions.

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 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 max tx amount

```
function setMaxTxAmount(uint256 maxTxAmount) external onlyOwner() {
        _maxTxAmount = maxTxAmount;
}
```

Contract owner can change the fees up to 100%

```
function setTaxFeePercent(uint256 taxFee) external onlyOwner() {
      _taxFee = taxFee;
}

function setLiquidityFeePercent(uint256 liquidityFee) external onlyOwner() {
      _liquidityFee = liquidityFee;
}
```

Contract owner can change marketingWalletAddress address and marketingDivisor

Current values:

marketingWalletAddress: 0x19cb40654fef74f84ca0c6fc3d83a6447846f38d

marketingDivisor: 3

```
function setMarketingAddress(address _marketingAddress) external onlyOwner() {
      marketingAddress = payable(_marketingAddress);
}

function setMarketingDivisor(uint256 divisor) external onlyOwner() {
      marketingDivisor = divisor;
}
.
.
.
function swapTokens(uint256 contractTokenBalance) private lockTheSwap {
      uint256 initialBalance = address(this).balance;
      swapTokensForEth(contractTokenBalance);
      uint256 transferredBalance = address(this).balance.sub(initialBalance);
      transferToAddressETH(marketingAddress, transferredBalance.div(_liquidityFee).mul(marketingDivisor));
}
```

Contract owner can change buyback settings

```
function setBuyBackEnabled(bool _enabled) public onlyOwner {
      buyBackEnabled = _enabled;
      emit BuyBackEnabledUpdated(_enabled);
}

function setBuybackUpperLimit(uint256 buyBackLimit) external onlyOwner() {
      buyBackUpperLimit = buyBackLimit * 10**18;
}
```

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;
}
```

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:	9%
Sell fees:	9%
Max TX:	1,000,000,000
Max Sell:	N/A

Honeypot Risk

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

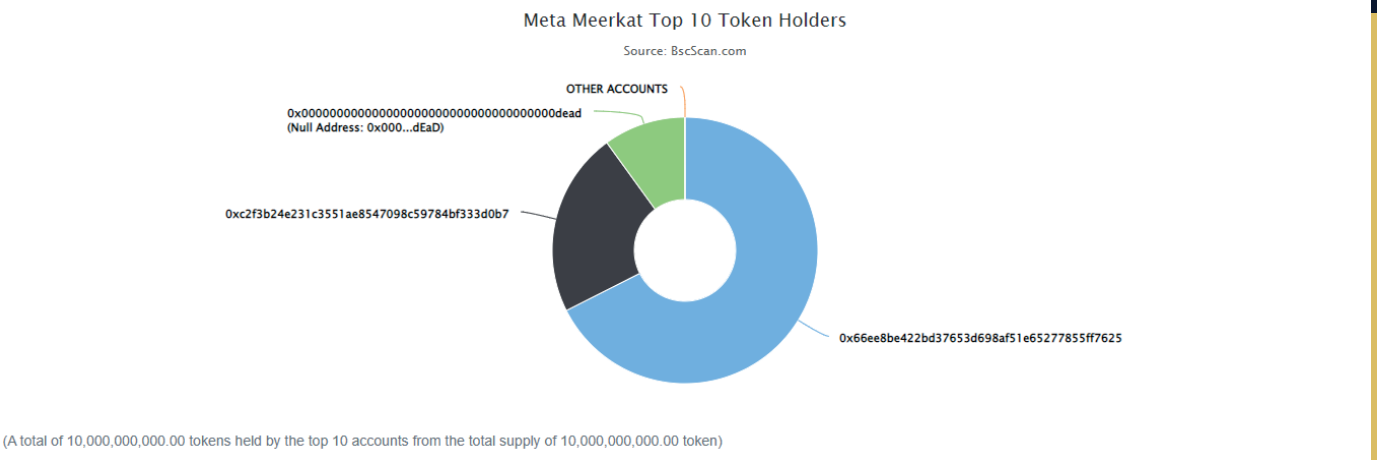
Rug Pull Risk

Liquidity:	N/A
Holders:	Clean

META MEERKAT TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

The top 10 holders collectively own 100.00% (10,000,000,000.00 Tokens) of Meta Meerkat

Token Total Supply: 10,000,000,000.00 Token | Total Token Holders: 3



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

Rank	Address	Quantity (Token)	Percentage
1	0x66ee8be422bd37653d698af51e65277855ff7625	6,755,400,000	67.5540%
2	0xc2f3b24e231c3551ae8547098c59784bf333d0b7	2,244,600,000	22.4460%
3	Null Address: 0x000...dEaD	1,000,000,000	10.0000%

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.