



SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT



MemeGold
\$MGLD

02/08/2023



TOKEN OVERVIEW

Fees

- Buy fees: 5%
- Sell fees: 5%

Fees privileges

- Can change buy fees up to 12% and sell fees up to 15%

Ownership

- Owned

Minting

- No mint function

Max Tx Amount / Max Wallet Amount

- Can change max tx amount (with threshold)

Blacklist

- Blacklist function not detected

Other privileges

- Can exclude / include from fees
-

TABLE OF CONTENTS

1

DISCLAIMER

2

INTRODUCTION

3

WEBSITE + SOCIALS

4-5

AUDIT OVERVIEW

6-9

OWNER PRIVILEGES

10

CONCLUSION AND ANALYSIS

11

TOKEN DETAILS

12

MGLD TOKEN ANALYTICS &
TOP 10 TOKEN HOLDERS

13

TECHNICAL DISCLAIMER



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

0x188592F3E031a6531A05b50deCAaF1C25adAb40F

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 **02/08/2023**



WEBSITE DIAGNOSTIC

<https://www.memegoldcoin.org/>



0-49



50-89



90-100



Performance



Accessibility



Best
Practices



SEO



Progressive
Web App

Socials



Twitter

<https://twitter.com/memegoldcoin>



Telegram

<https://t.me/memegoldcoin>

AUDIT OVERVIEW



Security Score



Static Scan

Automatic scanning for common vulnerabilities



ERC Scan

Automatic checks for ERC's conformance



High



Medium



Low



Optimizations



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't exclude an address from transactions
- Contract owner can exclude/include wallet from tax

```
function excludeFromFee(address account) public onlyOwner {
    require(!_isExcludedFromFee[account], "Account is already excluded");
    _isExcludedFromFee[account] = true;
}

function includeInFee(address account) public onlyOwner {
    require(_isExcludedFromFee[account], "Account is already included");
    _isExcludedFromFee[account] = false;
}
```

- Contract owner can enable/disable tax

```
function disableFees() public onlyOwner {
    prevLiqFee = _liquidityFee;
    prevTaxFee = _taxFee;
    prevDevFee = _devFee;
    prevSellFee = _sellTaxFee;
    _maxTxAmount = _tTotal;
    _liquidityFee = 0;
    _taxFee = 0;
    _devFee = 0;
    _sellTaxFee = 0;
    swapAndLiquifyEnabled = false;
}

function enableFees() public onlyOwner {
    _maxTxAmount = _tTotal;
    _liquidityFee = prevLiqFee;
    _taxFee = prevTaxFee;
    _devFee = prevDevFee;
    _sellTaxFee = prevSellFee;
    swapAndLiquifyEnabled = true;
}
```

- Contract owner can change swap settings

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

● Contract owner can change buy fees up to 12% and sell fees up to 15%

```
maxTaxFee = 4;  
maxLiqFee = 4;  
maxDevFee = 4;  
maxSellTaxFee = 7;
```

```
function setTaxFeePercent(uint256 taxFee) external onlyOwner() {  
    require(taxFee >= 0 && taxFee <= maxTaxFee, "taxFee out of range");  
    _taxFee = taxFee;  
    _previousTaxFee = _taxFee;  
}  
  
function setLiquidityFeePercent(uint256 liquidityFee) external onlyOwner() {  
    require(liquidityFee >= 0 && liquidityFee <= maxLiqFee, "liquidityFee out of range");  
    _liquidityFee = liquidityFee;  
    _previousLiquidityFee = _liquidityFee;  
}  
  
function setDevFeePercent(uint256 devFee) external onlyOwner() {  
    require(devFee >= 0 && devFee <= maxDevFee, "teamFee out of range");  
    _devFee = devFee;  
    _previousDevFee = _devFee;  
}  
  
function setSellTaxFeePercent(uint256 sellTaxFee) external onlyOwner() {  
    require(sellTaxFee >= 0 && sellTaxFee <= maxSellTaxFee, "taxFee out of range");  
    _sellTaxFee = sellTaxFee;  
    _previousSellFee = _sellTaxFee;  
}
```

● Contract owner can change _devWalletAddress address

Current value:

_devWalletAddress: 0x3101b782cda1631c9fb2cd1b3477abc20cf06f80

```
function setDevWalletAddress(address _addr) internal virtual {  
    if (!_isExcludedFromFee[_addr]) {  
        excludeFromFee(_addr);  
    }  
    _isdevWallet[_addr] = true;  
    _devWalletAddress = _addr;  
}  
  
function replaceDevWalletAddress(address _addr, address _newAddr) public onlyOwner {  
    require(_isdevWallet[_addr], "Wallet address not set previously");  
    require(!_isdevWallet[_newAddr], "Wallet address already set");  
    if (_isExcludedFromFee[_addr]) {  
        includeInFee(_addr);  
    }  
    _isdevWallet[_addr] = false;  
    setDevWalletAddress(_newAddr);  
}
```

● Contract owner can change max tx amount limitation (with threshold)

`minMxTxPercentage = 50;`

```
function setMaxTxPercent(uint256 maxTxPercent) external onlyOwner() {  
    require(maxTxPercent >= minMxTxPercentage && maxTxPercent <=100,"maxTxPercent out of range");  
    _maxTxAmount = _tTotal.mul(maxTxPercent).div(10**2);  
}
```

● 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 no HIGH issues during the first review.

TOKEN DETAILS

Details

Buy fees:	5%
Sell fees:	5%
Max TX:	1,000,000,000,000
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:	12,19% unlocked tokens



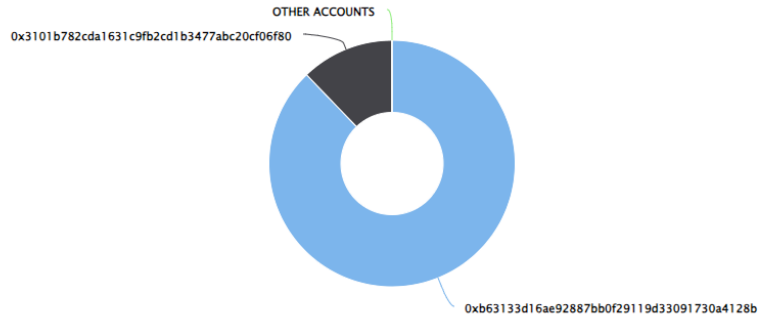
MGLD TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

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


Token Total Supply: 1,000,000,000,000.00 Token | Total Token Holders: 2

MemeGold Top 10 Token Holders

Source: BscScan.com



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

Rank	Address	Quantity (Token)	Percentage
1	 0xb63133d16ae92887bb0f29119d33091730a4128b	878,077,200,000	87.8077%
2	0x3101b782cda1631c9fb2cd1b3477abc20cf06f80	121,922,800,000	12.1923%

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.

