freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Meta Base

$MBAS

23/04/2023

TOKEN OVERVIEW

Fees

Buy fees:	1%
Sell fees:	5%

Fees privileges

- Can’t set or change fees

Ownership

- Owned

Minting

- No mint function

Max Tx Amount / Max Wallet Amount

- Can’t change max tx amount and / or max wallet amount

Blacklist

- Blacklist function not detected

Other privileges

- Can burn tokens from a particular wallet

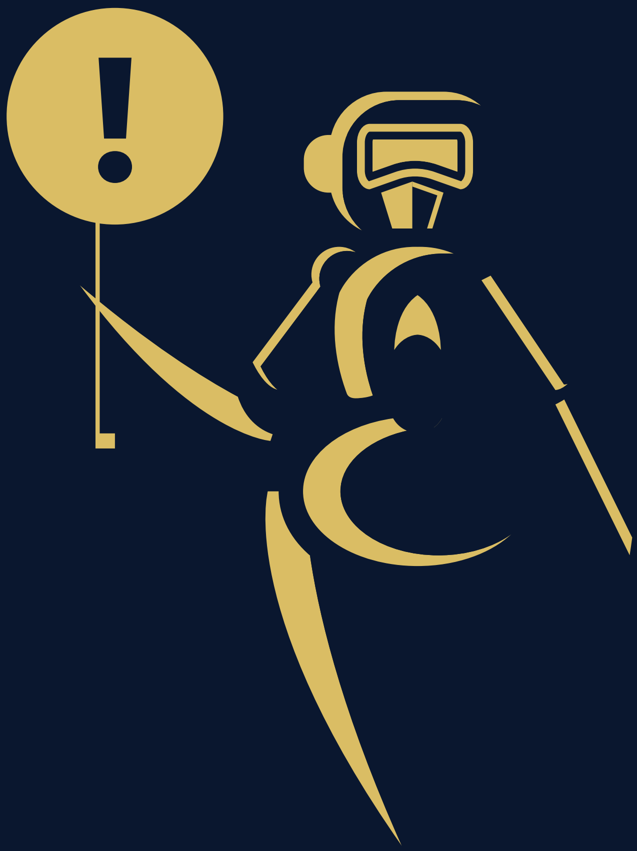
TABLE OF CONTENTS

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

0x094B109d635c34f24b1dC784b2b09F30ccf6408C

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 23/04/2023

AUDIT OVERVIEW

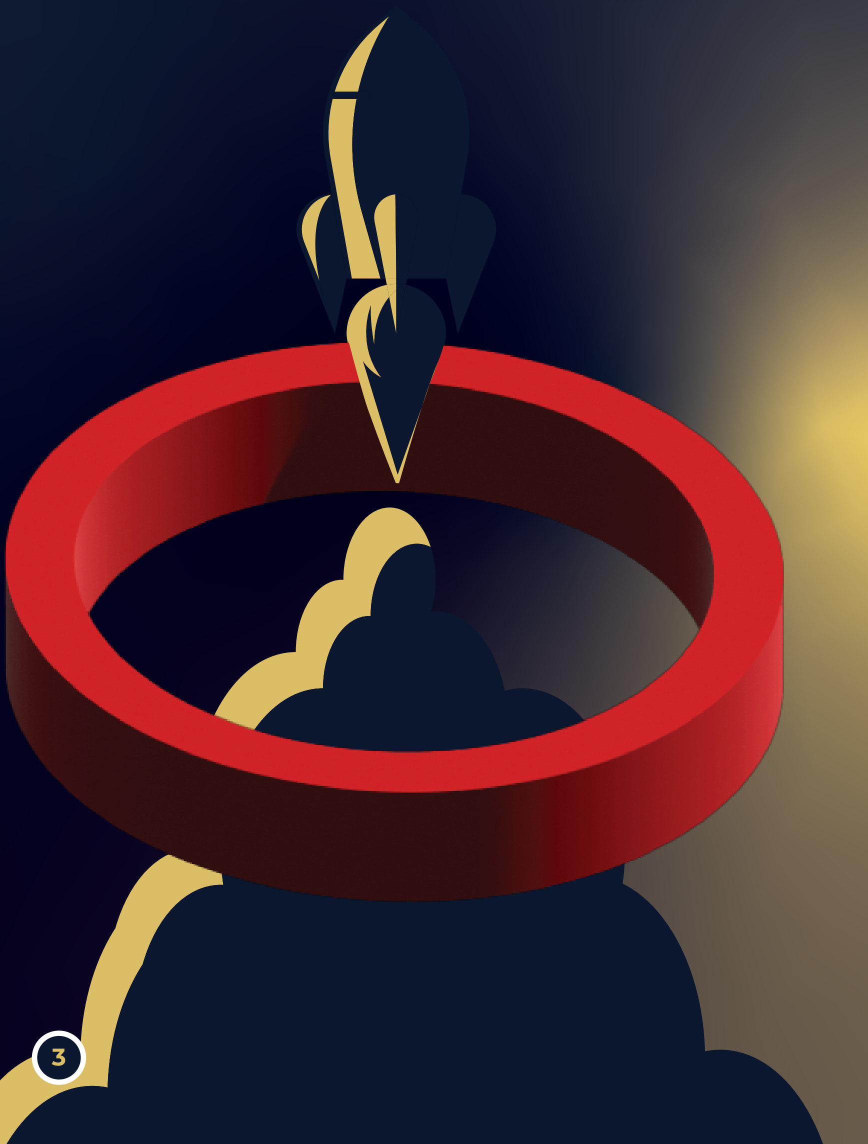
68

Security Score
HIGH RISK
Audit FAIL

93 **Static Scan**
Automatic scanning for common vulnerabilities

77 **ERC Scan**
Automatic checks for ERC's conformance

2	High
1	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't exclude an address from transactions

🟡 Contract owner can exclude/include wallet from tax

```
function setExcludeFee(address _address, bool _state) public onlyOwner {
        require(_address != address(0), "ERROR:ZERO_ADDRESS");
        excludeFee[_address] = _state;
}
```

🔴 Contract owner can burn tokens from a particular wallet

```
function burn(uint256 amount) public virtual {
        _burn(_msgSender(), amount);
}

function burnFrom(address account, uint256 amount) public virtual {
        _spendAllowance(account, _msgSender(), amount);
        _burn(account, amount);
}
```

🔴 Contract owner can change deadAddress address

Current value:

deadAddress: 0x000000000000000000000000000000000000dEaD

Changing the value of the deadAddress to another wallet address can have a significant impact. Any tokens sent to this new wallet address will be permanently destroyed.

```
function setDeadAddress(
        address _deadAddress,
        bool _isDaedAdd
    ) public onlyOwner {
        deadAddress[_deadAddress] = _isDaedAdd;
}

_transfer function line 1202
...
} else {
            // normal transfer
            if (deadAddress[to]) {
                _burn(from, amount);
            } else {
                _standardTransfer(from, to, amount);
            }
}
```

Contract owner can change operationAddress and burnFundAdd addresses

Current values:

operationAddress : 0x883d78eef43cba74dfc7c7ee1a381d5aa2b4f5bf

burnFundAdd : 0x7f4720e72e5d95d6825b16aafb0cfa01c024b78a

```
function setOperationAddress(address _address) public onlyOwner {
      require(_address != address(0), "ERROR:ZERO_ADDRESS");
      operationAddress = _address;
}

function setBurnFundAdd(address _burnFundAdd) public onlyOwner {
      require(_burnFundAdd != address(0), "ERROR:ZERO_ADDRESS");
      burnFundAdd = _burnFundAdd;
}
```

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);
}

function _transferOwnership(address newOwner) internal virtual {
      address oldOwner = _owner;
      _owner = newOwner;
      emit OwnershipTransferred(oldOwner, 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 2 HIGH issues during the first review.

TOKEN DETAILS

Details

Buy fees:	1%
Sell fees:	5%
Max TX:	N/A
Max Sell:	N/A

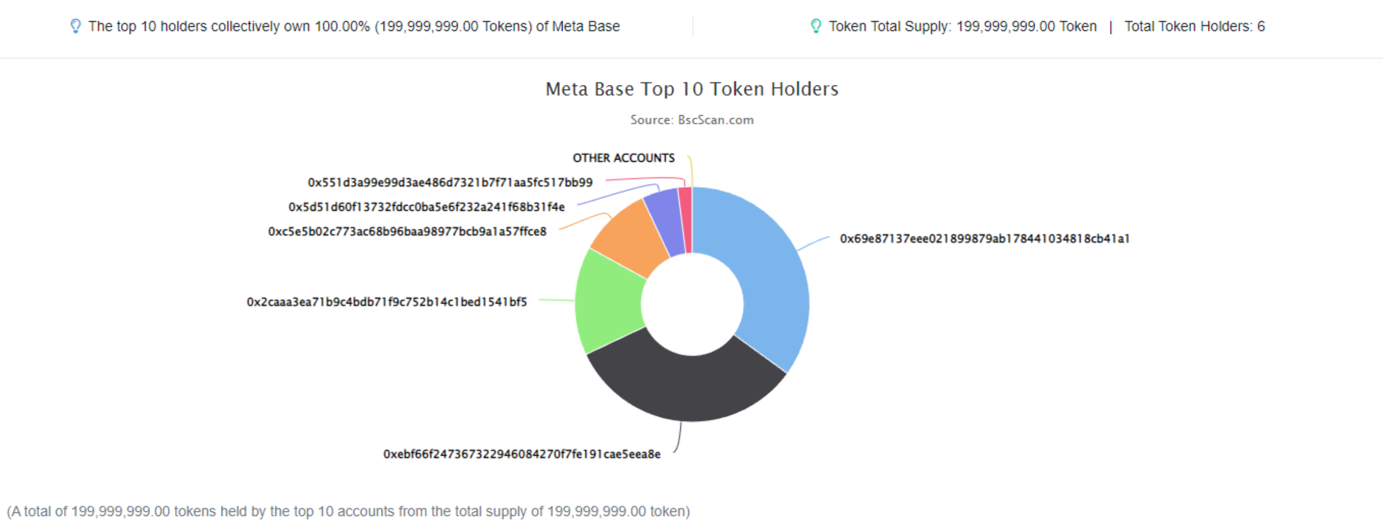
Honeypot Risk

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

Others

Liquidity:	N/A
Holders:	Clean

MBAS TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS



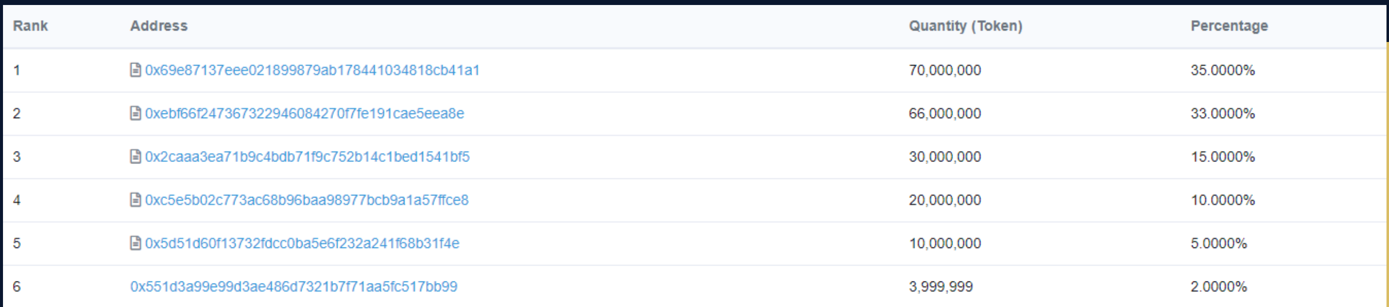
The top 10 holders collectively own 100.00% (199,999,999.00 Tokens) of Meta Base

Token Total Supply: 199,999,999.00 Token | Total Token Holders: 6

Meta Base Top 10 Token Holders

Source: BscScan.com

(A total of 199,999,999.00 tokens held by the top 10 accounts from the total supply of 199,999,999.00 token)



Rank	Address	Quantity (Token)	Percentage
1	0x69e87137eee021899879ab178441034818cb41a1	70,000,000	35.0000%
2	0xebf66f247367322946084270f7fe191cae5eea8e	66,000,000	33.0000%
3	0x2caaa3ea71b9c4bdb71f9c752b14c1bed1541bf5	30,000,000	15.0000%
4	0xc5e5b02c773ac68b96baa98977bcb9a1a57ffce8	20,000,000	10.0000%
5	0x5d51d60f13732fdcc0ba5e6f232a241f68b31f4e	10,000,000	5.0000%
6	0x551d3a99e99d3ae486d7321b7f71aa5fc517bb99	3,999,999	2.0000%

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.