freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Memenunity

$Memenunity

01/06/2022

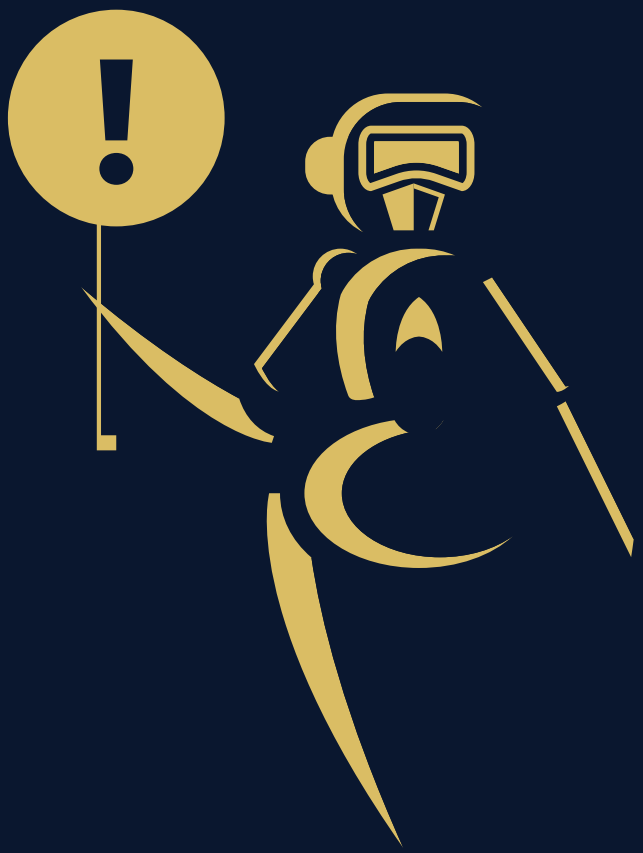
TABLE OF CONTENTS

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

0x5601c18Eaab27DE606d756609bf54e9f84A5df91

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 01/06/2022

AUDIT OVERVIEW

97

Security Score

97

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 an address from transactions

```
function blacklistAddress(address _address, bool _value) public authorized{
    isBlacklisted[_address] = _value;
}
```

Contract owner can exclude/include wallet from tax

```
function setIsFeeExempt(address holder, bool exempt) external authorized {
    isFeeExempt[holder] = exempt;
}
```

Contract owner can exclude/include wallet from tx limitations

```
function setIsTxLimitExempt(address holder, bool exempt) external authorized {
    isTxLimitExempt[holder] = exempt;
}
```

Contract owner can withdraw tokens from smart contract excluding Memenunity tokens

```
function transferForeignToken(address _token) public authorized {
    require(_token != address(this), "Can't let you take all native token");
    uint256 _contractBalance = IBEP20(_token).balanceOf(address(this));
    payable(marketingFeeReceiver).transfer(_contractBalance);
}
```

Contract owner can change swap settings

```
function setSwapBackSettings(bool _enabled, uint256 _amount) external authorized {
    swapEnabled = _enabled;
    swapThreshold = _amount;
}
```

Contract owner can change max wallet amount (with threshold)

```
function setMaxWallet(uint256 amount) external onlyOwner() {
    require(amount >= _totalSupply / 1000 );
    _maxWalletSize = amount;
}
```

Contract owner can change max tx amount (with threshold)

```
function setTxLimit(uint256 amount) external authorized {
    require(amount >= _totalSupply / 1000);
    _maxTxAmount = amount;
}
```

Contract owner can change fees up to 100%

```
function setFees(uint256 _liquidityFee, uint256 _teamFee, uint256 _marketingFee, uint256 _feeDenominator)
external authorized {
    liquidityFee = _liquidityFee;
    teamFee = _teamFee;
    marketingFee = _marketingFee;
    totalFee = _liquidityFee.add(_teamFee).add(_marketingFee);
    feeDenominator = _feeDenominator;
}
```

Contract owner can change marketingFeeReceiver and teamFeeReceiver addresses

Default values

marketingFeeReceiver : 0x8143c18571b4d21582f0290A7E6196015D00567C

teamFeeReceiver : 0x728888A30Fe10D8aF26AB7a8388E1f6876CB3792

```
function setFeeReceiver(address _marketingFeeReceiver, address _teamFeeReceiver) external authorized {
    marketingFeeReceiver = _marketingFeeReceiver;
    teamFeeReceiver = _teamFeeReceiver;
}
```

Contract owner can transfer ownership

```
function transferOwnership(address payable adr) public onlyOwner {
    owner = adr;
    authorizations[adr] = true;
    emit OwnershipTransferred(adr);
}
```

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:	6%
Sell fees:	7%
Max TX:	10,000,000,000
Max Sell:	N/A

Honeypot Risk

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

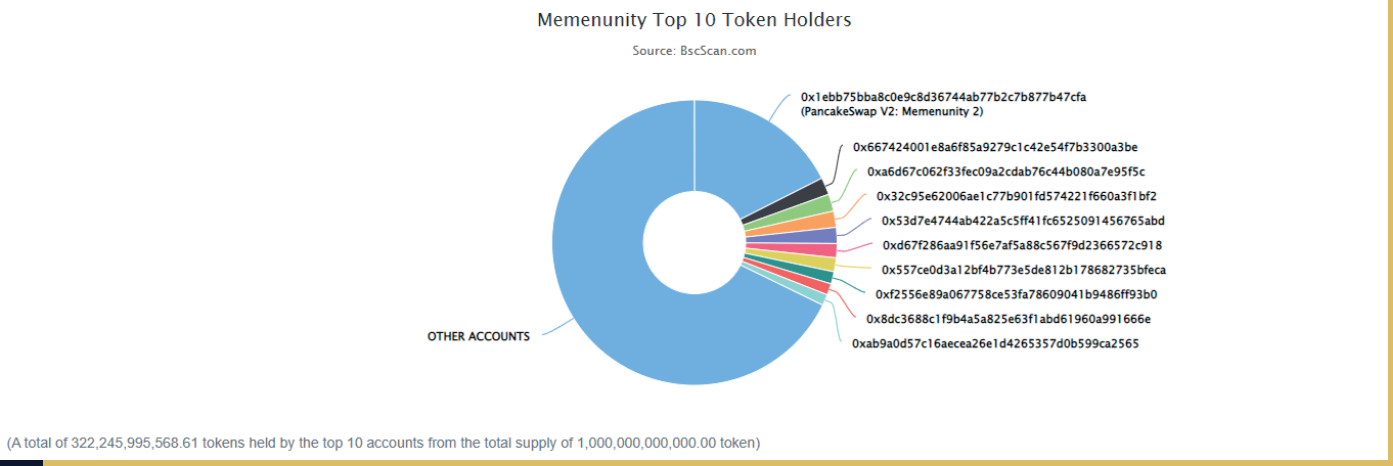
Rug Pull Risk

Liquidity:	N/A
Holders:	Clean

MEMENUNITY TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

The top 10 holders collectively own 32.22% (322,245,995,568.61 Tokens) of Memenunity

Token Total Supply: 1,000,000,000,000.00 Token | Total Token Holders: 421



(A total of 322,245,995,568.61 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	PancakeSwap V2: Memenunity 2	175,469,742,431.969524743	17.5470%
2	0x667424001e8a6f85a9279c1c42e54f7b3300a3be	19,624,804,894.668549138	1.9625%
3	0xa6d67c062f33fec09a2cdab76c44b080a7e95f5c	19,342,384,230.931445288	1.9342%
4	0x32c95e62006ae1c77b901fd574221f660a3f1bf2	18,520,390,621.834699383	1.8520%
5	0x53d7e4744ab422a5c5ff41fc6525091456765abd	18,198,995,236.29443113	1.8199%
6	0xd67f286aa91f56e7af5a88c567f9d2366572c918	15,995,021,091.869610814	1.5995%
7	0x557ce0d3a12bf4b773e5de812b178682735bfeca	15,818,883,033.887470333	1.5819%
8	0xf2556e89a067758ce53fa78609041b9486ff93b0	13,842,369,309.422296906	1.3842%
9	0x8dc3688c1f9b4a5a825e63f1abd61960a991666e	12,850,241,800	1.2850%
10	0xab9a0d57c16aecea26e1d4265357d0b599ca2565	12,583,162,917.736066302	1.2583%

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.