freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Meta Tindr

$TNDR

17/04/2022

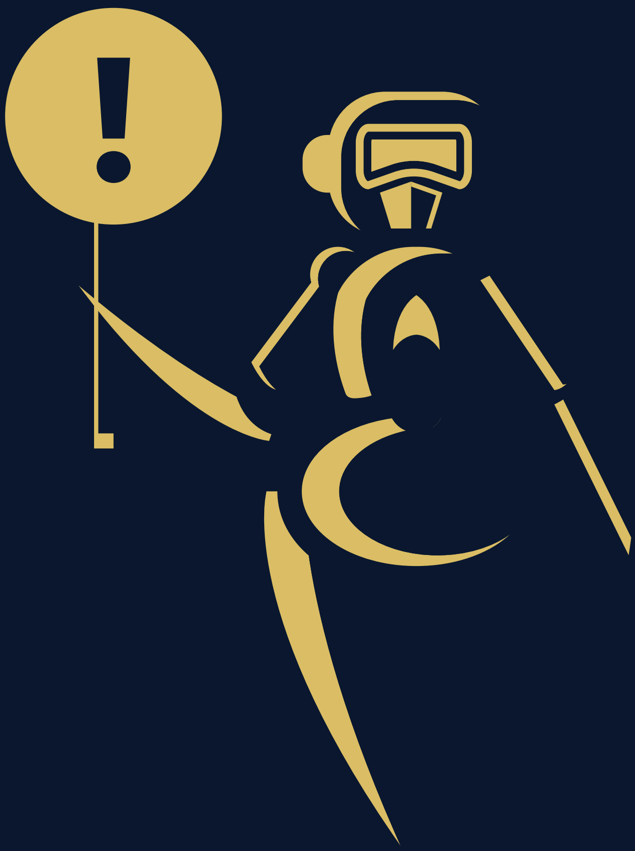
TABLE OF CONTENTS

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

0xD6b15F41c2b209215D67bCcB2FF4493d8ddd34ee

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 17/04/2022

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 mint tokens after initial contract deploy

Contract owner can't exclude an address from transactions

Contract owner can exclude/include wallet from rewards

```
function excludeFromReward(address account) public onlyOwner {
        // require(account != 0x7a250d5630B4cF539739dF2C5dAcb4c659F2488D, 'We can not exclude Uniswap router.');
        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 exclude/include wallet(s) from tax

```
function excludeFromFee(address account) public onlyOwner {
        _isExcludedFromFee[account] = true;
}

function includeInFee(address account) public onlyOwner {
        _isExcludedFromFee[account] = false;
}
```

Contract owner can change swap settings

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

Contract owner can change fees up to 25%

```
function setTaxFeePercent(uint256 taxFeeBps) external onlyOwner {
      _taxFee = taxFeeBps;
      require(
         _taxFee + _liquidityFee + _charityFee <= 10**4 / 4,
         "Total fee is over 25%"
      );
}

function setLiquidityFeePercent(uint256 liquidityFeeBps)
      external
      onlyOwner
   {
      _liquidityFee = liquidityFeeBps;
      require(
         _taxFee + _liquidityFee + _charityFee <= 10**4 / 4,
         "Total fee is over 25%"
      );
}
```

Contract owner can renounce ownership

```
function renounceOwnership() public virtual onlyOwner {
      _setOwner(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");
      _setOwner(newOwner);
}
```

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:	6%
Max TX:	N/A
Max Sell:	N/A

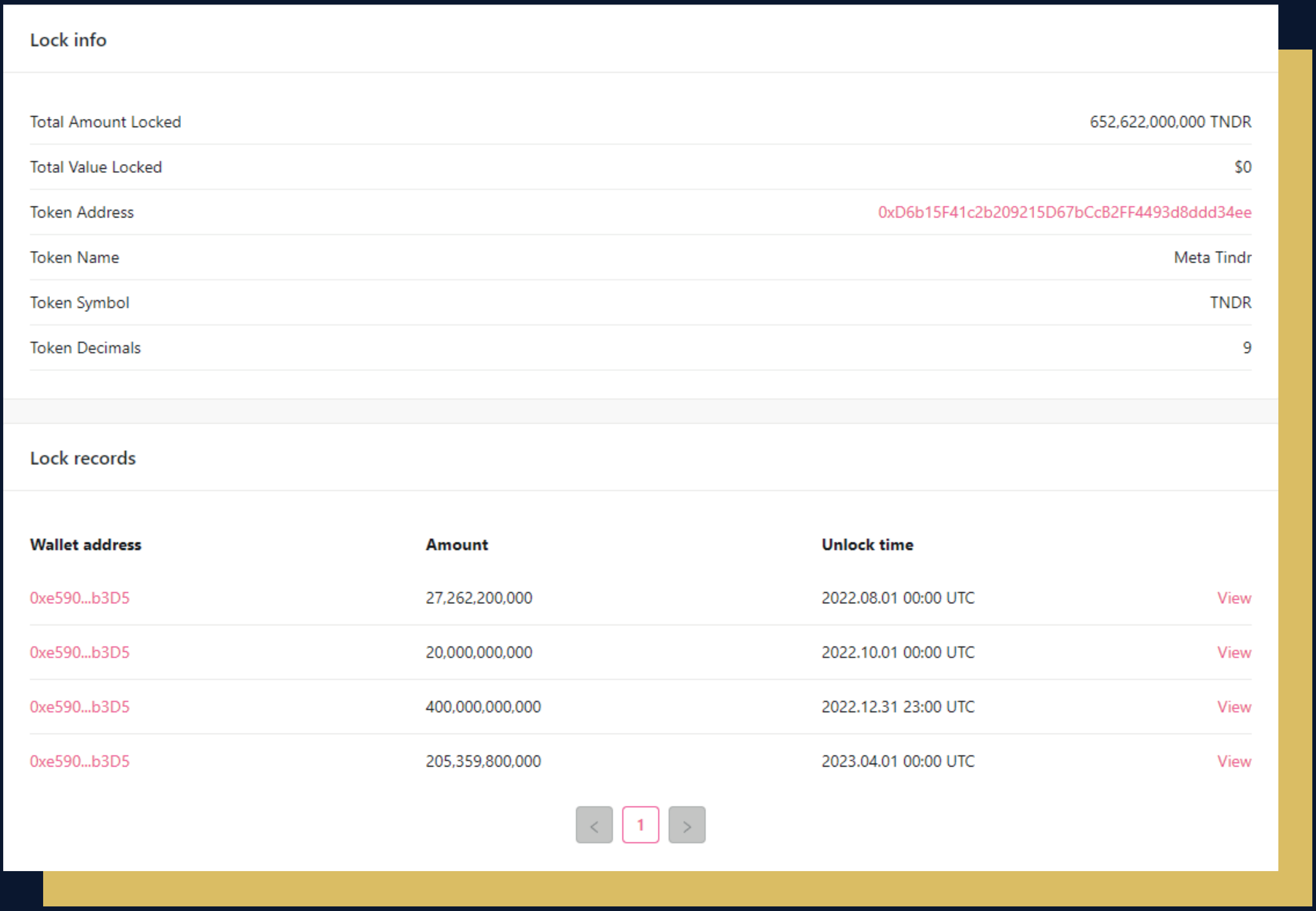
Honeypot Risk

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

Rug Pull Risk

Liquidity:	N/A
Holders:	Clean

META TINDR TOKEN LOCK RECORDS



Lock info

Total Amount Locked	652,622,000,000 TNDR
Total Value Locked	$0
Token Address	0xD6b15F41c2b209215D67bCcB2FF4493d8ddd34ee
Token Name	Meta Tindr
Token Symbol	TNDR
Token Decimals	9

Lock records

Wallet address	Amount	Unlock time	
0xe590...b3D5	27,262,200,000	2022.08.01 00:00 UTC	View
0xe590...b3D5	20,000,000,000	2022.10.01 00:00 UTC	View
0xe590...b3D5	400,000,000,000	2022.12.31 23:00 UTC	View
0xe590...b3D5	205,359,800,000	2023.04.01 00:00 UTC	View

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.