freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Tate Revolution

$TAREV

25/07/2023

TOKEN OVERVIEW

Fees

- Buy fees: N/A
- Sell fees: N/A

Fees privileges

- Can’t change / set fees

Ownership

- Owned

Minting

- No mint function

Max Tx Amount / Max Wallet Amount

- Can change max wallet amount (without threshold)

Blacklist

- Blacklist function detected

Other privileges

- Can burn tokens

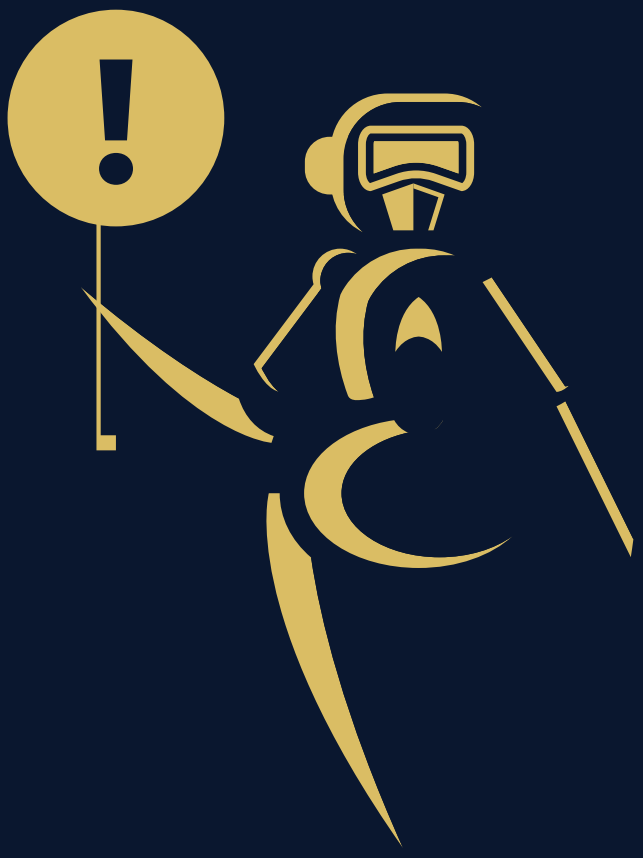
TABLE OF CONTENTS

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

0xfb3CCecc2183a786A851207dEB8d8945A361c9A6

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 25/07/2023

WEBSITE DIAGNOSTIC

https://www.tarev.vip/


0-49


50-89


90-100

Performance

Accessibility

Best Practices

SEO

Progressive Web App

Socials


Twitter

https://twitter.com/officialtarev


Telegram

https://t.me/tarevportal

AUDIT OVERVIEW

82

Security Score

92

Static Scan
Automatic scanning for common vulnerabilities

80

ERC Scan
Automatic checks for ERC's conformance

1 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 exclude an address from transactions

```
function blacklist(address _address, bool _isBlacklisting) external onlyOwner {
    blacklists[_address] = _isBlacklisting;
}
```

- Contract owner can change wallet limitations (without threshold)

Setting maxHoldingAmount to a very low value or minHoldingAmount to a very high value it may restrict users from performing certain actions

Current values:
maxHoldingAmount: 0;
minHoldingAmount: 0;

```
function setRule(bool _limited, address _uniswapV2Pair, uint256 _maxHoldingAmount, uint256 _minHoldingAmount) external onlyOwner {
    limited = _limited;
    uniswapV2Pair = _uniswapV2Pair;
    maxHoldingAmount = _maxHoldingAmount;
    minHoldingAmount = _minHoldingAmount;
}
```

- Contract owner can burn tokens

```
function _burn(address account, uint256 amount) internal virtual {
    require(account != address(0), "ERC20: burn from the zero address");

    _beforeTokenTransfer(account, address(0), amount);

    uint256 accountBalance = _balances[account];
    require(accountBalance >= amount, "ERC20: burn amount exceeds balance");
    unchecked {
      _balances[account] = accountBalance - amount;
    }
    _totalSupply -= amount;

    emit Transfer(account, address(0), amount);

    _afterTokenTransfer(account, address(0), amount);
}

function burn(uint256 value) external {
    _burn(msg.sender, value);
}
```

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 1 HIGH issues during the first review.

TOKEN DETAILS

Details

Buy fees:	N/A
Sell fees:	N/A
Max TX:	N/A
Max Sell:	N/A

Honeypot Risk

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

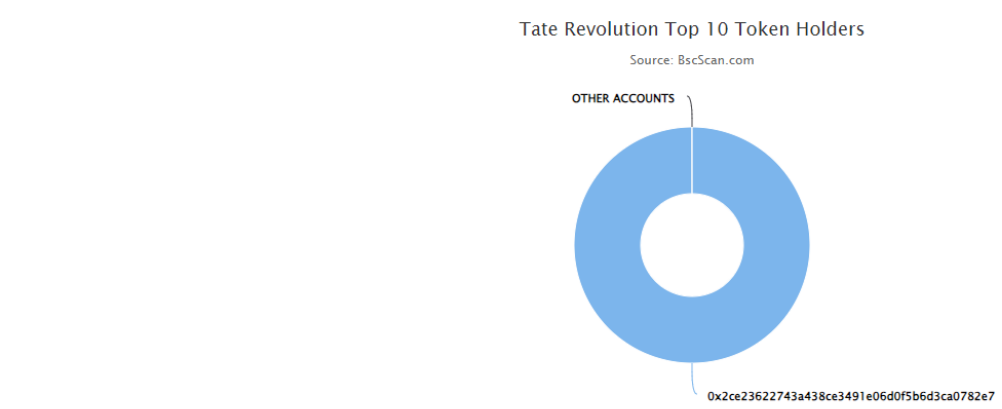
Rug Pull Risk

Liquidity:	N/A
Holders:	100% unlocked tokens

TAREV TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

The top 10 holders collectively own 100.00% (420,690,000,000,000.00 Tokens) of Tate Revolution

Token Total Supply: 420,690,000,000,000.00 Token | Total Token Holders: 1



(A total of 420,690,000,000,000.00 tokens held by the top 10 accounts from the total supply of 420,690,000,000,000.00 token)

Rank	Address	Quantity (Token)	Percentage
1	0x2ce23622743a438ce3491e06d0f5b6d3ca0782e7	420,690,000,000,000	100.0000%

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.