freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

ADSX

$ADX

19/10/2023

TOKEN OVERVIEW

Fees

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

Fees privileges

- Can change fees up to 10%

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 exclude / include from fees

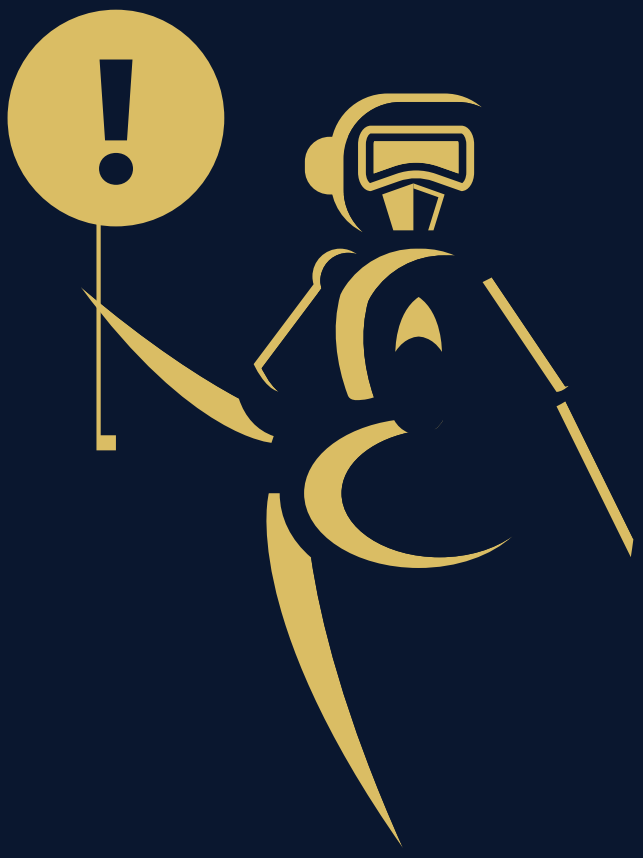
TABLE OF CONTENTS

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

0x1e58f542d65a220e7c7aa2259f59208207c41109

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 19/10/2023

WEBSITE DIAGNOSTIC

https://adsx.pro/

0-49 50-89 90-100

98	94	96	92	NA
Performance	Accessibility	Best Practices	SEO	Progressive Web App

Socials

Twitter

https://twitter.com/adsx_ads/

Telegram

https://t.me/AdsXOffical

AUDIT OVERVIEW

94

Security Score

96

Static Scan
Automatic scanning for common vulnerabilities

92

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	Medium
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 excludeFromFees(address account, bool excluded) external onlyOwner{
    require(_isExcludedFromFees[account] != excluded,"Account is already the value of 'excluded'");
    _isExcludedFromFees[account] = excluded;

    emit ExcludeFromFees(account, excluded);
}
```

- Contract owner can change swap settings

```
function setSwapTokensAtAmount(uint256 newAmount) external onlyOwner{
    require(newAmount > totalSupply() / 1_000_000, "SwapTokensAtAmount must be greater than 0.0001%
of total supply");
    swapTokensAtAmount = newAmount;

    emit SwapTokensAtAmountUpdated(swapTokensAtAmount);
}
```

- Contract owner can change marketingWallet and botDevWallet addresses

Current values:

marketingWallet: 0x033c78de4409EDCe3Ef830664ea326dd148fbD5B

botDevWallet: 0x033c78de4409EDCe3Ef830664ea326dd148fbD5B

```
function changeMarketingWallet(address _marketingWallet) external onlyOwner{
    require(_marketingWallet != marketingWallet,"Marketing wallet is already that address");
    require(_marketingWallet != address(0),"Marketing wallet cannot be the zero address");
    marketingWallet = _marketingWallet;

    emit MarketingWalletChanged(marketingWallet);
}

function changebotDevWallet(address _botDevWallet) external onlyOwner{
    require(_botDevWallet != botDevWallet,"Marketing wallet is already that address");
    require(_botDevWallet != address(0),"Marketing wallet cannot be the zero address");
    botDevWallet = _botDevWallet;

    emit botDevWalletChanged(botDevWallet);
}
```

Contract owner can change fees up to 10%

maxFee = 10;

```
function updateBuyFees(uint256 _marketingFeeOnBuy, uint256 _botDevFeeOnBuy) external onlyOwner {
        marketingFeeOnBuy = _marketingFeeOnBuy;
        botDevFeeOnBuy = _botDevFeeOnBuy;

        _totalFeesOnBuy   = marketingFeeOnBuy + botDevFeeOnBuy;

        require(_totalFeesOnBuy + _totalFeesOnSell <= maxFee, "Total Fees cannot exceed the maximum");

        emit UpdateBuyFees(marketingFeeOnBuy, botDevFeeOnBuy);
    }

    function updateSellFees(uint256 _marketingFeeOnSell, uint256 _botDevFeeOnSell) external onlyOwner {
        marketingFeeOnSell = _marketingFeeOnSell;
        botDevFeeOnSell = _botDevFeeOnSell;

        _totalFeesOnSell   = marketingFeeOnSell + botDevFeeOnSell;

        require(_totalFeesOnBuy + _totalFeesOnSell <= maxFee, "Total Fees cannot exceed the maximum");

        emit UpdateSellFees(marketingFeeOnSell, botDevFeeOnSell);
    }
```

Contract owner has ability to retrieve tokens held by the contract

Native tokens excluded

```
function claimStuckTokens(address token) external onlyOwner {
        require(token != address(this), "Owner cannot claim contract's balance of its own tokens");
        if (token == address(0x0)) {
            payable(msg.sender).sendValue(address(this).balance);
            return;
        }
        IERC20 ERC20token = IERC20(token);
        uint256 balance = ERC20token.balanceOf(address(this));
        ERC20token.transfer(msg.sender, balance);
    }
```

Contract owner can transfer ownership

```
function transferOwnership(address newOwner) public virtual onlyOwner {
    require(newOwner != address(0), 'Ownable: new owner is the zero address');
    emit OwnershipTransferred(_owner, newOwner);
    _owner = newOwner;
}
```

Contract owner can renounce ownership

```
function renounceOwnership() public virtual onlyOwner {
    emit OwnershipTransferred(_owner, address(0));
    _owner = 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:	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:	100% unlocked tokens

ADX TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

The top 10 holders collectively own 100.00% (100,000,000.00 Tokens) of ADSX

Token Total Supply: 100,000,000.00 Token | Total Token Holders: 1

(A total of 100,000,000.00 tokens held by the top 10 accounts from the total supply of 100,000,000.00 token)

Rank	Address	Quantity (Token)	Percentage
1	0x033c78...148fbD5B	100,000,000	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.