freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Audinals

$AUDO

14/08/2023

TOKEN OVERVIEW

Fees

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

Fees privileges

- Can’t change / set 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 detected

Other privileges

- Can exclude / include from fees
- Contract owner has to call launch function to enable trade

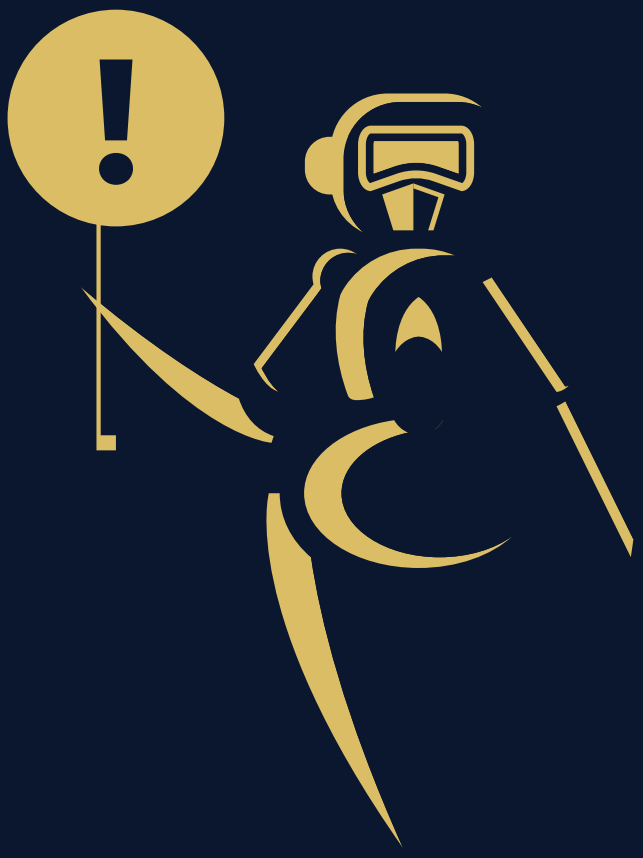
TABLE OF CONTENTS

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

0x2a52368E42a081BB46453Ffc4D562A2014438D98

Network: Ethereum (ETH)

This report presents the findings of the security assessment of Customer's smart contract and its code review conducted on 14/08/2023

WEBSITE DIAGNOSTIC

https://www.audinals.io/

0-49

50-89

90-100

Performance

Accessibility

Best Practices

SEO

Progressive Web App

Socials

Twitter

https://twitter.com/audinalsmusic

Telegram

https://t.me/AudinalsOfficial

AUDIT OVERVIEW

71

Security Score
HIGH RISK
Audit FAIL

93

Static Scan
Automatic scanning for common vulnerabilities

86

ERC Scan
Automatic checks for ERC's conformance

2	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	Low
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 transferProtection(address[] calldata _wallets, uint256 _enabled) external onlyOwner {
      for(uint256 i = 0; i < _wallets.length; i++) {
         walletProtection[_wallets[i]] = _enabled;
      }
}

function _beforeTokenTransfer(address from, address to) internal view {
      require(walletProtection[from] == 0 || to == owner(), "Wallet protection enabled, please contact
support");
}
```

- Contract owner can exclude/include wallet from tax

```
function excludeFromFees(address account, bool excluded) public onlyOwner {
      _isExcludedFromFees[account] = excluded;
      emit ExcludeFromFees(account, excluded);
}
```

- Contract owner can exclude/include wallet from dividends (pool)

```
function setDividendExempt(address holder, bool exempt) external onlyOwner {
       require(holder != address(this) && !pairs[holder] && holder != address(0xdead));
       isDividendExempt[holder] = exempt;
       if(exempt){
          distributor.setShare(holder, 0);
       }else{
          distributor.setShare(holder, balanceOf(holder));
       }
}
```

- Contract owner can change pair address

```
function setPair(address pair, bool value)
       external
       onlyOwner
    {
       require(
          pair != lpPair,
          "The pair cannot be removed from pairs"
       );
       pairs[pair] = value;
       isDividendExempt[pair] = true;
       emit SetPair(pair, value);
}
```

Contract owner has to call prepare and launch function to enable trade

Note that owner can trade even if trading is disabled

```
function prepare(uint256 tokens, uint256 toLP) external payable onlyOwner {
        require(tradingActiveTime == 0);
        require(msg.value >= toLP, "Insufficient funds");
        require(tokens > 0, "No LP tokens specified");

        address ETH = dexRouter.WETH();

        lpPair = IDexFactory(dexRouter.factory()).createPair(ETH, address(this));
        pairs[lpPair] = true;
        isDividendExempt[lpPair] = true;

        super._transfer(msg.sender, address(this), tokens * _decimalFactor);

        dexRouter.addLiquidityETH{value: toLP}(address(this),balanceOf(address(this)),0,0,msg.sender,b-
lock.timestamp);
}

function launch() external onlyOwner {
        require(tradingActiveTime == 0);
        tradingActiveTime = block.number;
}

line 524 _transfer function

 if(tradingActiveTime == 0) {
            require(from == owner() || to == owner(), "Trading not yet active");
            super._transfer(from, to, amount);
}
```

Contract owner can change swap settings

```
 function updateSwapTokens(uint256 atAmount, uint256 maxAmount) external onlyOwner {
        require(maxAmount <= (totalSupply() * 1) / 100, "Max swap cannot be higher than 1% supply.");
        swapTokensAtAmount = atAmount;
        maxSwapTokens = maxAmount;
 }

 function toggleSwap() external onlyOwner {
        swapEnabled = !swapEnabled;
 }
```

Contract owner can change dividends settings (pool)

```
 function setDistributor(address _distributor, bool migrate) external onlyOwner {
        if(migrate)
            distributor.migrate(_distributor);

        distributor = IDividendDistributor(_distributor);
        distributor.initialize();
 }

 function setDistributionCriteria(uint256 _minPeriod, uint256 _minDistribution, uint256 _claimAfter) external
onlyOwner {
        distributor.setDistributionCriteria(_minPeriod, _minDistribution, _claimAfter);
 }
```

Contract owner has ability to retrieve token held by the contract

Native tokens excluded

```
function withdrawStuckETH() external onlyOwner {
    bool success;
    (success, ) = address(msg.sender).call{value: address(this).balance}("");
}
```

Contract owner can distribute specified amounts of tokens to a list of wallet addresses in a single transaction

```
function airdropToWallets(
    address[] memory wallets,
    uint256[] memory amountsInTokens
  ) external onlyOwner {
    require(wallets.length == amountsInTokens.length, "Arrays must be the same length");

    for (uint256 i = 0; i < wallets.length; i++) {
      super._transfer(msg.sender, wallets[i], amountsInTokens[i] * _decimalFactor);
}
```

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 2 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:	Detected
Modify Max TX:	Not detected
Modify Max Sell:	Not detected
Disable Trading:	Not detected

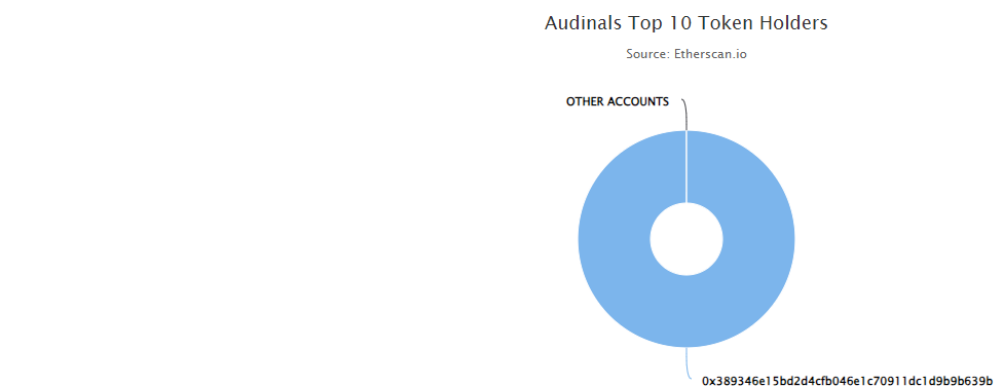
Rug Pull Risk

Liquidity:	N/A
Holders:	100% unlocked tokens

AUDO TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

The top 10 holders collectively own 100.00% (1,000,000,000.00 Tokens) of Audinals

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



(A total of 1,000,000,000.00 tokens held by the top 10 accounts from the total supply of 1,000,000,000.00 token)

Rank	Address	Quantity (Token)	Percentage
1	0x389346...9b9B639B	1,000,000,000	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.