freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

EuroFootball AI

$EFBAI

06/03/2023

TOKEN OVERVIEW

Fees

- Buy fees: 4%
- Sell fees: 6%

Fees privileges

- Can change buy fees up to 25% and sell feel up to 25%

Ownership

- Owned

Minting

- No mint function

Max Tx Amount / Max Wallet Amount

- Can change max tx amount and max wallet amount (with threshold)

Blacklist

- Blacklist function not detected

Other privileges

- Can exclude / include from fees

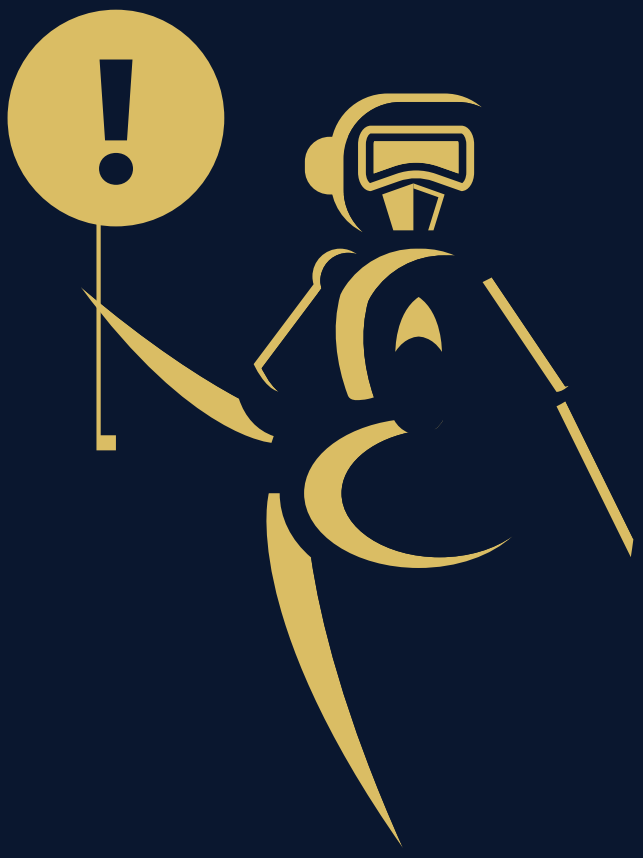
TABLE OF CONTENTS

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

0x8C8f52E1B08333EDF558e41FC252baA87f118296

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 06/03/2023

AUDIT OVERVIEW

92

Security Score

98

Static Scan
Automatic scanning for common vulnerabilities

90

ERC Scan
Automatic checks for ERC's conformance

0 High

3 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(s) from tax

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

- Contract owner can exclude/include wallet from tx limitations

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

- Contract owner can change wallet limitations (with threshold)

```
function setMaxWallet_base1000(uint256 maxwalletPercentage_base1000) external onlyOwner {
    require(_maxWalletSize >=_totalSupply.mul(2).div(1000),"Cannot set max wallet below .2%");
    _maxWalletSize = (_totalSupply * maxwalletPercentage_base1000 ) / 1000;
}
```

- Contract owner can change tx limitations (with threshold)

```
function setMaxTxPercent_base1000(uint256 maxTXPercentage_base1000) external onlyOwner() {
    require(_maxTxAmount >=_totalSupply.mul(2).div(1000),"Cannot set max tx below .2%");
    _maxTxAmount = (_totalSupply * maxTXPercentage_base1000 ) / 1000;
}
```

- Contract owner can change swap settings

```
function setSwapBackSettings(bool _enabled, uint256 _amountS, uint256 _amountL, bool _alternate)
external onlyOwner {
    alternateSwaps = _alternate;
    claimingFees = _enabled;
    smallSwapThreshold = _amountS;
    largeSwapThreshold = _amountL;
    swapThreshold = smallSwapThreshold;
}
```

- Contract owner has to call enableTrading function to enable trade

```
function enableTrading () public onlyOwner {
    tradingOpen = true;
}
```

Contract owner can change buy fees up to 25% and sell fees up to 25%

```
function changeFees(uint256 _liquidityFeeBuy, uint256 _reflectionFeeBuy, uint256 _marketingFeeBuy,
uint256 _TeamFeeBuy, uint256 _feeDenominator,
   uint256 _liquidityFeeSell, uint256 _reflectionFeeSell, uint256 _marketingFeeSell, uint256 _TeamFeeSell)
external onlyOwner {
      liquidityFeeBuy = _liquidityFeeBuy;
      reflectionFeeBuy = _reflectionFeeBuy;
      marketingFeeBuy = _marketingFeeBuy;
      TeamFeeBuy = _TeamFeeBuy;
      totalFeeBuy = liquidityFeeBuy.add(reflectionFeeBuy).add(marketingFeeBuy).add(TeamFeeBuy);

      liquidityFeeSell = _liquidityFeeSell;
      reflectionFeeSell = _reflectionFeeSell;
      marketingFeeSell = _marketingFeeSell;
      TeamFeeSell = _TeamFeeSell;
      totalFeeSell = liquidityFeeSell.add(reflectionFeeSell).add(marketingFeeSell).add(TeamFeeSell);

      feeDenominator = _feeDenominator;

      require(totalFeeBuy <= feeDenominator / 4, "Buy Fees can not be more than 25%");
      require(totalFeeSell <= feeDenominator / 4, "Sell Fees can not be more than 25%");
}
```

Contract owner can change marketingFeeReceiver, TeamFeeReceiver and autoLiquidityReceiver addresses

Default values:

marketingFeeReceiver: 0x566165F2763BF8288bdebe51a638998Dc6A45730

TeamFeeReceiver: 0x2C693A81bA5Dc30B1D2CC050DC6f866C0fF692A5

autoLiquidityReceiver: 0x2C693A81bA5Dc30B1D2CC050DC6f866C0fF692A5

```
function setFeeReceivers(address _marketingFeeReceiver, address _liquidityReceiver, address _TeamFeeRe-
ceiver) external onlyOwner {
      marketingFeeReceiver = _marketingFeeReceiver;
      TeamFeeReceiver = _TeamFeeReceiver;
      autoLiquidityReceiver = _liquidityReceiver;
}
```

Contract owner can withdraw tokens from smart contract (EFBAI not excluded)

```
function clearBalance() external {
     (bool success,) = payable(autoLiquidityReceiver).call{value: address(this).balance, gas: 30000}("");
      require(success);
}

function clearForeignToken(address tokenAddress, uint256 tokens) public returns (bool) {
      require(isTxLimitExempt[msg.sender]);
   if(tokens == 0){
        tokens = IERC20(tokenAddress).balanceOf(address(this));
      }
      return IERC20(tokenAddress).transfer(msg.sender, tokens);
}
```

- 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:	4%
Sell fees:	6%
Max TX:	1,000,000,000
Max Sell:	N/A

Honeypot Risk

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

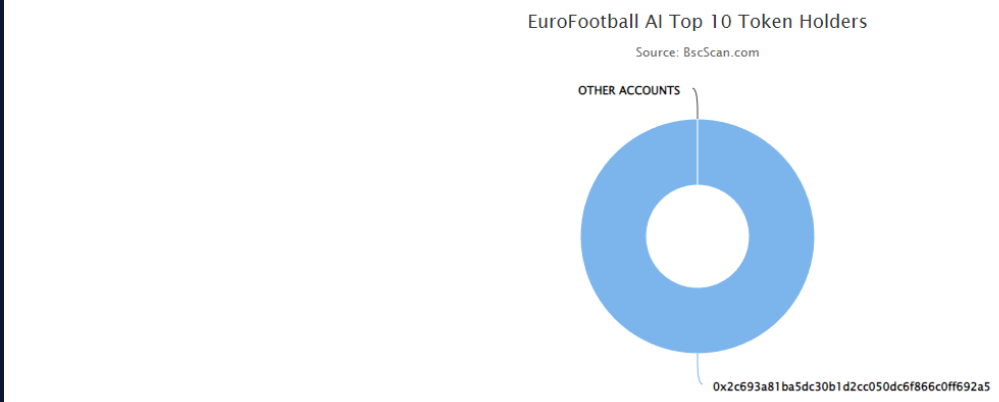
Rug Pull Risk

Liquidity:	N/A
Holders:	Clean

EFBAI TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

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

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	0x2c693a81ba5dc30b1d2cc050dc6f866c0ff692a5	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.