Cyberscope

Audit Report

Nut2Earn

August 2022

SHA256 be6ab611f1f014748fb083a7bf8687248c6c8e689fba2b3abe61c1dd3f9cde83

Audited by © cyberscope

Table of Contents

Source Files

Filename	SHA256
contract.sol	be6ab611f1f014748fb083a7bf8687248c6c8e689fba2b3abe61c1dd3f9cde83

Audit Updates

Initial Audit	31st July 2022
Corrected phase 1	1st August 2022
Corrected phase 2	2st August 2022

Contract Analysis

Critical Medium Minor Pass

Severity	Code	Description
●	ST	Contract Owner is not able to stop or pause transactions
●	OCTD	Contract Owner is not able to transfer tokens from specific address
●	OTUT	Owner Transfer User's Tokens
●	ELFM	Contract Owner is not able to increase fees more than a reasonable percent (25%)
●	ULTW	Contract Owner is not able to increase the amount of liquidity taken by dev wallet more than a reasonable percent
●	MT	Contract Owner is not able to mint new tokens
●	BT	Contract Owner is not able to burn tokens from specific wallet
●	BC	Contract Owner is not able to blacklist wallets from selling

Contract Diagnostics

Critical Medium Informative

Severity	Code	Description
	MTS	Manipulate Total Supply
	L04	Conformance to Solidity Naming Conventions

MTS - Manipulate Total Supply

Criticality	informative
Location	contract.sol#L791

Description

Contract is able to manipulate total supply. Since it is a rebase token that is automatically distributed to holders, the total supply will increase with each distribution. This change will have a direct impact on the token price and marketcap.

```
function coreRebase(int256 supplyDelta) private returns (uint256) {
    uint256 epoch = block.timestamp;
    if (supplyDelta == 0) {
        emit LogRebase(epoch, _totalSupply);
        return _totalSupply;
    } else {
        if ((_totalSupply.add(uint256(supplyDelta))) >= MAX_SUPPLY) {
        // in case the rebase will cause the supply to pass MAX_SUPPLY,
autorebase will be turned off & rebase will not happen.
        autoRebase = false;
        emit LogRebase(epoch, _totalSupply);
        return _totalSupply;
    } else {
        _totalSupply = _totalSupply.add(uint256(supplyDelta));
        }
    }
    _gonsPerFragment = TOTAL_GONS.div(_totalSupply);
    nextRebase = epoch + rebaseFrequency;
    emit LogRebase(epoch, _totalSupply);
    return _totalSupply;
}
```

Recommendation

The contract owner should carefully manage the adjustment of the circulating supply (increases or decreases), according to the token's price fluctuations.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L324,331,332,333,419

Description

Solidity defines a naming convention that should be followed. Rule exceptions:

- Allow constant variable name/symbol/decimals to be lowercase.
- Allow _ at the beginning of the mixed_case match for private variables and unused parameters.

```
_gonSwapThreshold
ZeroWalletAddress
DeadWalletAddress
BusdToken
feeDenominator
```

Recommendation

Follow the Solidity naming convention.
https://docs.soliditylang.org/en/v0.4.25/style-guide.html#naming-conventions.

Contract Functions

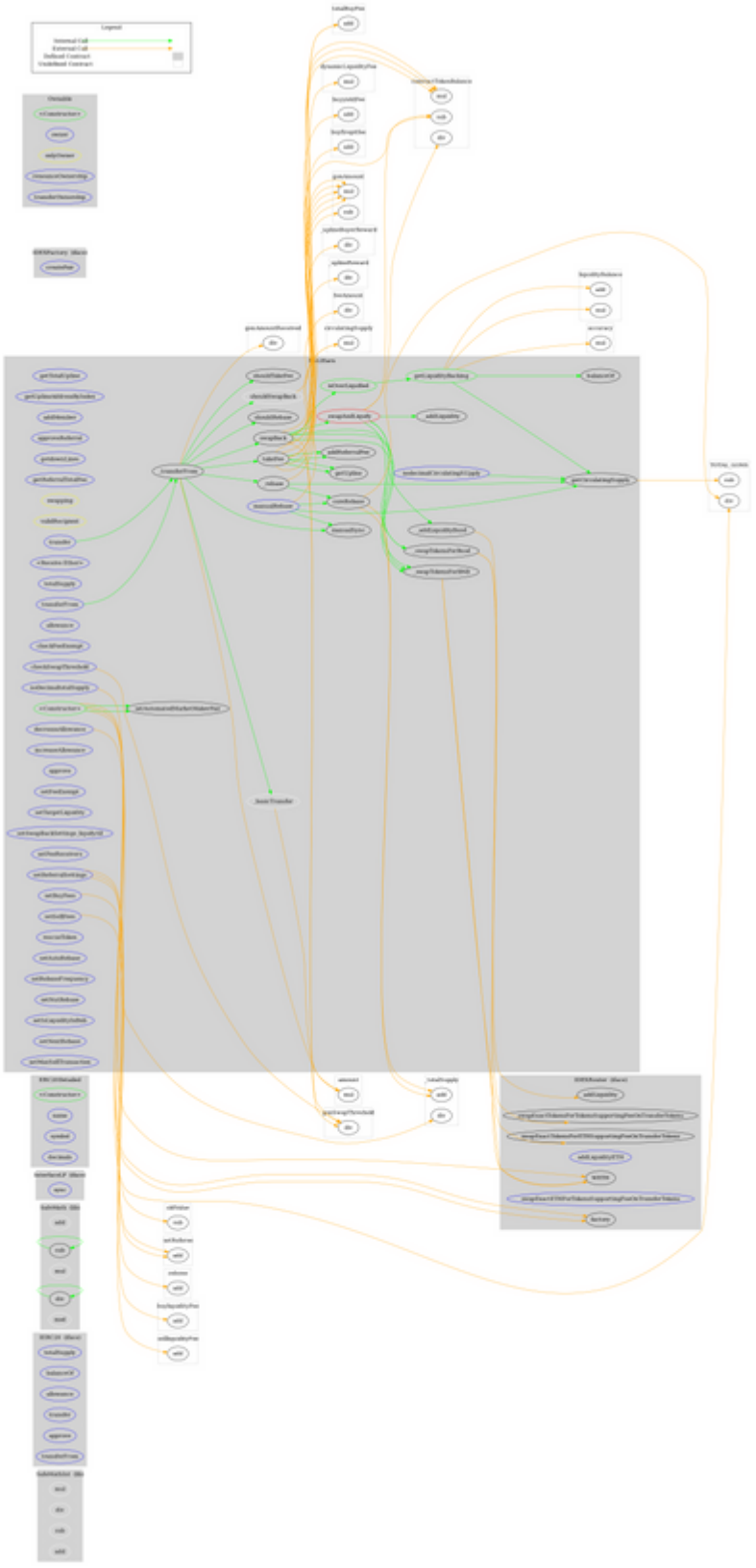
Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
SafeMathInt	Library			
	mul	Internal		
	div	Internal		
	sub	Internal		
	add	Internal		
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	allowance	External		-
	transfer	External	✓	-
	approve	External	✓	-
	transferFrom	External	✓	-
SafeMath	Library			
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
	mod	Internal		
InterfaceLP	Interface			
	sync	External	✓	-
ERC20Detailed	Implementation	IERC20		
	<Constructor>	Public	✓	-
	name	External		-

	symbol	External		-
	decimals	External		-
IDEXRouter	Interface			
	factory	External		-
	weth	External		-
	addLiquidity	External	✓	-
	addLiquidityETH	External	Payable	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
IDEXFactory	Interface			
	createPair	External	✓	-
Ownable	Implementation			
	<Constructor>	Public	✓	-
	owner	External		-
	renounceOwnership	External	✓	onlyOwner
	transferOwnership	External	✓	onlyOwner
Nut2Earn	Implementation	ERC20Detailed, Ownable		
	getTotalUpline	External		-
	getUplineAddressByIndex	External		-
	addMember	External	✓	onlyOwner
	approveReferral	External	✓	-
	getUpline	Public		-
	getdownLines	External		-
	addReferralFee	Public	✓	-
	getReferralTotalFee	External		-
	<Constructor>	Public	✓	ERC20Detailed
	<Receive Ether>	External	Payable	-

	totalSupply	External		-
	noDecimaltotalSupply	External		-
	nodecimalCirculatingSUpply	External		-
	allowance	External		-
	balanceOf	Public		-
	checkFeeExempt	External		-
	checkSwapThreshold	External		-
	shouldRebase	Internal		
	shouldTakeFee	Internal		
	shouldSwapBack	Internal		
	getCirculatingSupply	Public		-
	getLiquidityBacking	Public		-
	isOverLiquified	Public		-
	manualSync	Public	✓	-
	transfer	External	✓	validRecipient
	_basicTransfer	Internal	✓	
	_transferFrom	Internal	✓	
	transferFrom	External	✓	validRecipient
	_swapAndLiquify	Private	✓	
	_addLiquidity	Private	✓	
	_addLiquidityBusd	Private	✓	
	_swapTokensForBNB	Private	✓	
	_swapTokensForBusd	Private	✓	
	swapBack	Internal	✓	swapping
	takeFee	Internal	✓	
	decreaseAllowance	External	✓	-
	increaseAllowance	External	✓	-
	approve	External	✓	-
	_rebase	Private	✓	
	coreRebase	Private	✓	
	manualRebase	External	✓	onlyOwner
	setAutomatedMarketMakerPair	Public	✓	onlyOwner
	setFeeExempt	External	✓	onlyOwner
	setTargetLiquidity	External	✓	onlyOwner
	setSwapBackSettings	External	✓	onlyOwner

	setFeeReceivers	External	✓	onlyOwner
	setReferralSettings	External	✓	onlyOwner
	setBuyFees	External	✓	onlyOwner
	setSellFees	External	✓	onlyOwner
	rescueToken	External	✓	onlyOwner
	setAutoRebase	External	✓	onlyOwner
	setRebaseFrequency	External	✓	onlyOwner
	setNutRebase	External	✓	onlyOwner
	setIsLiquidityInBnb	External	✓	onlyOwner
	setNextRebase	External	✓	onlyOwner
	setMaxSellTransaction	External	✓	onlyOwner

Contract Flow



Summary

Nut2Earn is an interesting project that has a friendly and growing community. The Smart Contract uses a total supply manipulation business model to provide value for the holders. The contract Owner can access some admin functions that can not be used in a malicious way to disturb the users' transactions. There is also a limit of max 25% fees.

Disclaimer

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment.

Cyberscope team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed.

The Cyberscope team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Cyberscope receive a payment to manipulate those results or change the awarding badge that we will be adding in our website.

Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token.

The Cyberscope team disclaims any liability for the resulting losses.

About Cyberscope

Coinscope audit and K.Y.C. service has been rebranded to Cyberscope.

Coinscope is the leading early coin listing, voting and auditing authority firm. The audit process is analyzing and monitoring many aspects of the project. That way, it gives the community a good sense of security using an informative report and a generic score.

Cyberscope and Coinscope are aiming to make crypto discoverable and efficient globally. They provide all the essential tools to assist users draw their own conclusions.

The Cyberscope team

https://www.cyberscope.io