Cyberscope

Audit Report

Ring Wrap Token

March 2022

Type	BEP20
Network	BSC
Address	0x59AE8c783eBCe3CC68ccE32C427128101fa4C405
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	WrappedRingERC20
Compiler Version	v0.7.5+commit.eb77ed08
Optimization	200 runs
Licence	Unknown
Explorer	https://bscscan.com/token/0x59AE8c783eBCe3CC68ccE32C427128101fa4C405
Symbol	wRING
Decimals	18
Total Supply	-
Source	Address.sol, Counters.sol, ERC20.sol, IERC20.sol, IPancakeSwapFactory.sol, IPancakeSwapRouter.sol, IRing.sol, Ownable.sol, Ring.sol, SafeERC20.sol, SafeMath.sol, **WrappedRing.sol**
Domain	ringfi.io

Audit Updates

Initial Audit	14th March 2022
Corrected	

Token Wrapping Feature

The wRing contract implements the standard ERC functionality enriched with wrap and unwrap feature. The wRing tokens are pegged to the value of RING and the value of the wRing token is moved proportionally to the value of the RING crypto. If the user wants to return to their RING asset they simply trade their wRing tokens back to the smart contract and exit with the corresponding value of the RING tokens.

- `wrap()` receives RING and gives wRing
- `unwrap()` receives wRing and gives RING

The wRing rate is fixed to the “index” variant that is provided by the RING contract. The index value is fixed in the RING contract for 100000 tokens from the initial supply.

For instance, let’s assume that a user holds 10 RING. According to the index rate the `wrap()` function will yield 1 wRing token. On the other hand, if the user holds 1 wRing tokens, the `unwrap()` function will yield 10 RING tokens.

In this Audit we will focus on the Wrapping contract.

Contracts Balance Concern

The wRing contract provides the functionality of converting a RING token with wRing. The wRing does not provide any guarantee regarding the price rate of these two tokens. That means that the underneath price of these two tokens is independent. For instance:

1. User holds 30 Ring, the price of Ring is X and the price of wRing is Y
2. User converts 10 Ring for 1 wRing. So, the user holds 20 Ring and 1 wRing.
3. User sells 10 Ring. The price of the Ring is X - X1 and the price of wRing is Y.
4. In this state the price of x will vary from Y and there is not an in-chain functionality that fixes the price misalignment.

The before mentioned flow is just a concern about the business logic of the token wrapping functionality.

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 Minor

Severity	Code	Description
	CO	Code Optimization
	MC	Missing Check
	L01	Public Function could be Declared External
	L02	State Variables could be Declared Constant
	L04	Conformance to Solidity Naming Conventions
	L11	Unnecessary Boolean equality
	L13	Divide before Multiply Operation

CO - Code Optimization

Criticality	minor
Location	contract.sol#L41

Description

There are code segments that could be optimized. A segment may be optimized so that it becomes a smaller size, consumes less memory, executes more rapidly, or performs fewer operations.

The RING address is initialized once in the constructor. Then it is used either as an IERC20 or IRING interface. The IRING inherits the IERC20 interface.

```
RING = _RING;

IERC20(RING);

IRING(RING);
```

Recommendation

The RING type could be declared as IRING, so there will no need for wrapping the RING variable in all th occustances.

MC - Missing Check

Criticality	minor
Location	contract.sol#L57,72

Description

The contract is processing variables that have not properly sanitized and checked that they form the proper shape. These variables may produce vulnerability issues.

The wrap and unwrap functions are proceeding on the corresponding functionality without checking if the “sender” or the “contract” holds the necessary funds. For instance, the unwrap function pre-requirements are:

- User should hold more than “_amount” tokens
- The RING contract balance should hold more than “value” tokens

```
function unwrap(uint256 _amount) external returns (uint256) {
    require(live == true, "wRING: unwrapping disabled");

    _burn(msg.sender, _amount);

    uint256 value = wRINGToRING(_amount);
    IERC20(RING).transfer(msg.sender, value);
    return value;
}
```

Recommendation

The contract should properly check the variables according to the required specifications

L01 - Public Function could be Declared External

Criticality	minor
Location	WrappedRing.sol#L184,195,206,214

Description

Public functions that are never called by the contract should be declared external to save gas.

```
toggleWhitelist
setPairFee
setFeeReceivers
setLiveStatus
```

Recommendation

Use the external attribute for functions never called from the contract

L02 - State Variables could be Declared Constant

Criticality	minor
Location	WrappedRing.sol#L31,22,24,25,26,23

Description

Constant state variables should be declared constant to save gas.

```
treasuryFee
supplyControlFee
sellFee
ringRiskFreeFundFee
liquidityFee
feeDenominator
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	WrappedRing.sol#L57,72,87,96,184,195,206,214,16,20 and 5 more

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.

```
_isFeeExempt
_pairWithFee
RING
_addr
_supplyControl
_ringRiskFreeFund
_treasuryFund
_autoLiquidityFund
_live
...
```

Recommendation

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

L11 - Unnecessary Boolean equality

Criticality	minor
Location	WrappedRing.sol#L57,72

Description

The comparison to boolean constants is redundant. Boolean constants can be used directly and do not need to be compared to true or false.

```
require(bool,string)(live == true,wRING: unwrapping disabled)
require(bool,string)(live == true,wRING: wrapping disabled)
```

Recommendation

Remove the equality to the boolean constant.

L13 - Divide before Multiply Operation

Criticality	minor
Location	WrappedRing.sol#L162

Description

Performing divisions before multiplications may cause lose of prediction.

```
_balances[supplyControl] =
_balances[supplyControl].add(amount.div(feeDenominator).mul(supplyControlFee))
_balances[ringRiskFreeFund] =
_balances[ringRiskFreeFund].add(amount.div(feeDenominator).mul(ringRiskFreeFundF
ee))
_balances[treasuryFund] =
_balances[treasuryFund].add(amount.div(feeDenominator).mul(treasuryFee))
_balances[autoLiquidityFund] =
_balances[autoLiquidityFund].add(amount.div(feeDenominator).mul(liquidityFee))
feeAmount = amount.div(feeDenominator).mul(_totalFee)
```

Recommendation

The multiplications should be prior to the divisions.

Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
Address	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	_functionCallWithValue	Private	✓	
	functionStaticCall	Internal		
	functionStaticCall	Internal		
	functionDelegateCall	Internal	✓	
	functionDelegateCall	Internal	✓	
	_verifyCallResult	Private		
	addressToString	Internal		
Counters	Library			
	current	Internal		
	increment	Internal	✓	
	decrement	Internal	✓	
ERC20	Implementation	IERC20		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-

	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
IERC2612Permit	Interface			
	permit	External	✓	-
	nonces	External		-
ERC20Permit	Implementation	ERC20, IERC2612Permit		
	<Constructor>	Public	✓	-
	permit	Public	✓	-
	nonces	Public		-
IERC20	Interface			
	name	External		-
	symbol	External		-
	decimals	External		-
	totalSupply	External		-
	balanceOf	External		-
	allowance	External		-
	approve	External	✓	-
	transfer	External	✓	-
	transferFrom	External	✓	-
IERC20Mintable	Interface			
	mint	External	✓	-
	mint	External	✓	-

IPancakeSwap Factory	Interface			
	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-
IPancakeSwap Router	Interface			
	factory	External		-
	WETH	External		-
	addLiquidity	External	✓	-
	addLiquidityETH	External	Payable	-
	removeLiquidity	External	✓	-
	removeLiquidityETH	External	✓	-
	removeLiquidityWithPermit	External	✓	-
	removeLiquidityETHWithPermit	External	✓	-
	swapExactTokensForTokens	External	✓	-
	swapTokensForExactTokens	External	✓	-
	swapExactETHForTokens	External	Payable	-
	swapTokensForExactETH	External	✓	-
	swapExactTokensForETH	External	✓	-
	swapETHForExactTokens	External	Payable	-
	quote	External		-
	getAmountOut	External		-
	getAmountIn	External		-
	getAmountsOut	External		-
	getAmountsIn	External		-
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-

	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
IRING	Interface	IERC20		
	getCirculatingSupply	External		-
	gonsForBalance	External		-
	balanceForGons	External		-
	returnMsgSender	External		-
	index	External		-
Ownable	Implementation			
	<Constructor>	Public	✓	-
	owner	Public		-
	isOwner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
SafeMathInt	Library			
	mul	Internal		
	div	Internal		
	sub	Internal		
	add	Internal		
	abs	Internal		
SafeMath	Library			
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		

	mod	Internal		
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	allowance	External		-
	transfer	External	✓	-
	approve	External	✓	-
	transferFrom	External	✓	-
IPancakeSwap Pair	Interface			
	name	External		-
	symbol	External		-
	decimals	External		-
	totalSupply	External		-
	balanceOf	External		-
	allowance	External		-
	approve	External	✓	-
	transfer	External	✓	-
	transferFrom	External	✓	-
	DOMAIN_SEPARATOR	External		-
	PERMIT_TYPEHASH	External		-
	nonces	External		-
	permit	External	✓	-
	MINIMUM_LIQUIDITY	External		-
	factory	External		-
	token0	External		-
	token1	External		-
	getReserves	External		-
	price0CumulativeLast	External		-
	price1CumulativeLast	External		-
	kLast	External		-
	mint	External	✓	-
	burn	External	✓	-
	swap	External	✓	-

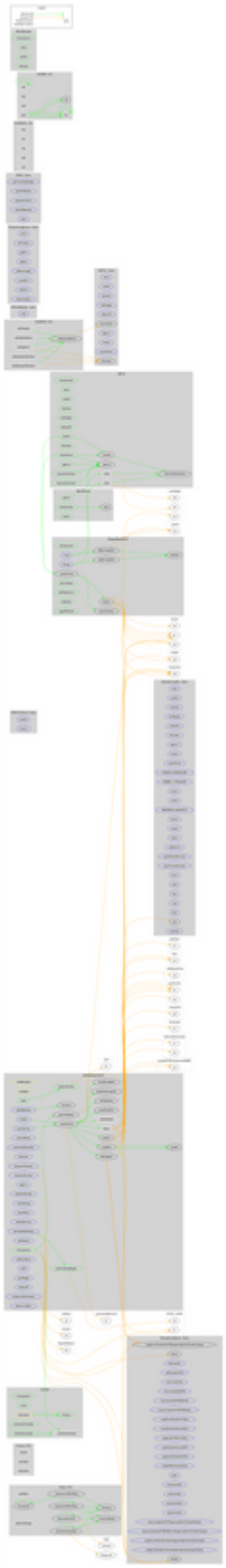
	skim	External	✓	-
	sync	External	✓	-
	initialize	External	✓	-
IPancakeSwap Router	Interface			
	factory	External		-
	WETH	External		-
	addLiquidity	External	✓	-
	addLiquidityETH	External	Payable	-
	removeLiquidity	External	✓	-
	removeLiquidityETH	External	✓	-
	removeLiquidityWithPermit	External	✓	-
	removeLiquidityETHWithPermit	External	✓	-
	swapExactTokensForTokens	External	✓	-
	swapTokensForExactTokens	External	✓	-
	swapExactETHForTokens	External	Payable	-
	swapTokensForExactETH	External	✓	-
	swapExactTokensForETH	External	✓	-
	swapETHForExactTokens	External	Payable	-
	quote	External		-
	getAmountOut	External		-
	getAmountIn	External		-
	getAmountsOut	External		-
	getAmountsIn	External		-
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
IPancakeSwap Factory	Interface			

	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-
Ownable	Implementation			
	<Constructor>	Public	✓	-
	owner	Public		-
	isOwner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
ERC20Detailed	Implementation	IERC20		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
TestNetRingContract1	Implementation	ERC20Detailed, Ownable		
	<Constructor>	Public	✓	ERC20Detailed Ownable
	rebase	Internal	✓	
	transfer	External	✓	validRecipient
	transferFrom	External	✓	validRecipient
	_basicTransfer	Internal	✓	
	_transferFrom	Internal	✓	
	takeFee	Internal	✓	
	addLiquidity	Internal	✓	swapping
	swapBack	Internal	✓	swapping

	withdrawAllToTreasury	External	✓	swapping onlyOwner
	shouldTakeFee	Internal		
	shouldRebase	Internal		
	shouldAddLiquidity	Internal		
	shouldSwapBack	Internal		
	setAutoRebase	External	✓	onlyOwner
	setAutoAddLiquidity	External	✓	onlyOwner
	allowance	External		-
	decreaseAllowance	External	✓	-
	increaseAllowance	External	✓	-
	approve	External	✓	-
	checkFeeExempt	External		-
	getCirculatingSupply	Public		-
	isNotInSwap	External		-
	manualSync	External	✓	-
	setFeeReceivers	External	✓	onlyOwner
	getLiquidityBacking	External		-
	setWhitelist	External	✓	onlyOwner
	setBotBlacklist	External	✓	onlyOwner
	setPairAddress	External	✓	onlyOwner
	setLP	External	✓	onlyOwner
	totalSupply	External		-
	balanceOf	External		-
	isContract	Internal		
	gonsForBalance	Public		-
	balanceForGons	Public		-
	index	Public		-
	<Receive Ether>	External	Payable	-
SafeERC20	Library			
	safeTransfer	Internal	✓	
	safeTransferFrom	Internal	✓	
	safeApprove	Internal	✓	
	safeIncreaseAllowance	Internal	✓	
	safeDecreaseAllowance	Internal	✓	

	_callOptionalReturn	Private	✓	
SafeMath	Library			
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
	sqrrt	Internal		
WrappedRingERC20	Implementation	ERC20, Ownable		
	<Constructor>	Public	✓	ERC20 Ownable
	wrap	External	✓	-
	unwrap	External	✓	-
	wRINGToRING	Public		-
	RINGTowRING	Public		-
	shouldTakeFee	Internal		
	transfer	Public	✓	-
	transferFrom	Public	✓	-
	_transferFrom	Internal	✓	
	takeFee	Internal	✓	
	setLiveStatus	Public	✓	onlyOwner
	setFeeReceivers	Public	✓	onlyOwner
	setPairFee	Public	✓	onlyOwner
	toggleWhitelist	Public	✓	onlyOwner

Contract Flow



Domain Info

Domain Name	ringfi.io
Registry Domain ID	b213828d5e2045f99811904dfc9d8ec7-DONUTS
Creation Date	2022-03-02T11:13:19Z
Updated Date	2022-03-07T11:13:59Z
Registry Expiry Date	2023-03-02T11:13:19Z
Registrar WHOIS Server	whois.namecheap.com
Registrar URL	https://www.namecheap.com/
Registrar	NameCheap, Inc.
Registrar IANA ID	1068

The domain has been created 12 days before the creation of the audit. It will expire in 12 months.

There is no public billing information, the creator is protected by the privacy settings.

Summary

The Smart Contract analysis reported no compiler error or critical issues. 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 max fees limit of 14% in buys and 16% in sales. The contract incarnates a token wrapping functionality that does not affect the transactions. The scope of this audit focuses on the WrappedRing.sol file.

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 provides all the essential tools to assist users draw their own conclusions.

The Cyberscope team

https://www.cyberscope.io