Cyberscope

Audit Report

LUCKY RABBIT SPIN

December 2022

Type	BEP20
Network	BSC
Address	0x5878ADeA653b2f9148ba31beA7ed2F031D7603E6
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	LUCKYRABBITSPIN
Compiler Version	v0.8.4+commit.c7e474f2
Optimization	200 runs
Licence	None
Explorer	https://bscscan.com/token/0x5878ADeA653b2f9148ba31beA7ed2F031D7603E6
Symbol	$LRS
Decimals	9
Total Supply	100,000,000,000
Domain	luckyrabbit.games

Source Files

Filename	**SHA256**
contract.sol	c9d9d6a227c52959b21ecd190b1ba6246b289bca0e1c09067cda733bd8b4e227

Audit Updates

Initial Audit	7th December 2022
Corrected	

Contract Analysis

🔴 Critical 🟠 Medium ⚪ Minor / Informative 🔵 Pass

Severity	Code	Description	Status
🔵	ST	Stops Transactions	Passed
⚪	OCTD	Transfers Contract's Tokens	Unresolved
🔵	OTUT	Transfers User's Tokens	Passed
🔵	ELFM	Exceeds Fees Limit	Passed
⚪	ULTW	Transfers Liquidity to Team Wallet	Unresolved
🔵	MT	Mints Tokens	Passed
🔵	BT	Burns Tokens	Passed
🔵	BC	Blacklists Addresses	Passed

OCTD - Transfers Contract's Tokens

Criticality	Minor / Informative
Status	Unresolved

Description

The contract owner has the authority to claim all the balance of the contract. The owner may take advantage of it by calling the `manualswap` function.

```
function manualswap() external {
    require(_msgSender() == _developmentAddress || _msgSender() ==
_marketingAddress || _msgSender() == owner());
    uint256 contractBalance = balanceOf(address(this));
    swapTokensForEth(contractBalance);
}
```

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. That risk can be prevented by temporarily locking the contract or renouncing ownership.

ULTW - Transfers Liquidity to Team Wallet

Criticality	minor / informative
Location	contract.sol#L387
Status	Unresolved

Description

The contract owner has the authority to transfer funds without limit to the team wallet. These funds have been accumulated from fees collected from the contract. The owner may take advantage of it by calling the `manualsend` method.

```
function manualsend() external {
    require(_msgSender() == _developmentAddress || _msgSender() ==
_marketingAddress || _msgSender() == owner());
    uint256 contractETHBalance = address(this).balance;
    sendETHToFee(contractETHBalance);
}
```

Recommendation

The contract could embody a check for the maximum amount of funds that can be swapped. Since a huge amount may volatile the token's price.

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. That risk can be prevented by temporarily locking the contract or renouncing ownership.

Contract Diagnostics

🔴 Critical 🟠 Medium ● Minor / Informative

Severity	Code	Description	Status
●	PTRP	Potential Transfer Revert Propagation	Unresolved
●	DDP	Decimal Division Precision	Unresolved
●	RSML	Redundant SafeMath Library	Unresolved
●	L02	State Variables could be Declared Constant	Unresolved
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L05	Unused State Variable	Unresolved

PTRP - Potential Transfer Revert Propagation

Criticality	critical
Location	contract.sol#L296
Status	Unresolved

Description

The contract sends funds to a marketingWallet and a developmentWallet as part of the transfer flow. These addresses can either be a wallet address or a contract. If the address is a contract then it may revert from incoming payment. As a result, the error will propagate to the token's contract and revert the transfer.

```
function sendETHToFee(uint256 amount) private {
    _developmentAddress.transfer(amount.div(2));
    _marketingAddress.transfer(amount.div(2));
}
```

Recommendation

The contract should tolerate the potential revert from the underlying contracts when the interaction is part of the main transfer flow. This could be archived by not allowing set contract addresses or by sending the funds in a non-revertable way.

DDP - Decimal Division Precision

Criticality	minor / informative
Location	contract.sol#L297,298
Status	Unresolved

Description

The calculated value is the result of a division. Since Solidity has not floating types, then the result of a division may miss the decimals precision. As a result, the splitted shares will not have the exact precision and some funds may not be calculated as expected.

```
_developmentAddress.transfer(amount.div(2));
_marketingAddress.transfer(amount.div(2));
```

Recommendation

The contract could calculate the subtraction of the divided funds in the last calculation in order to avoid the division rounding issue.

RSML - Redundant SafeMath Library

Criticality	minor / informative
Location	contract.sol#L59
Status	Unresolved

Description

The Solidity versions that are greater than or equal to 0.8.0 do not need the use of SafeMath Library. The usage of the SafeMath library produces unnecessary additional gas.

```
library SafeMath {
...
}
```

Recommendation

The team is advised to remove the SafeMath library as it is safe to do math operations without it.

L02 - State Variables could be Declared Constant

Criticality	minor / informative
Location	contract.sol#L99
Status	Unresolved

Description

Constant state variables should be declared constant to save gas.

```
_previousOwner
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor / informative
Location	contract.sol#L305,151,306,311,153,138,152,40,318,404
Status	Unresolved

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.

```
tokensRescued
_name
_amount
devAddressUpdated
_tokenAddr
_decimals
_tTotal
_symbol
WETH
...
```

Recommendation

Follow the Solidity naming convention.
https://docs.soliditylang.org/en/v0.8.17/style-guide.html#naming-conventions.

L05 - Unused State Variable

Criticality	minor / informative
Location	contract.sol#L99,133
Status	Unresolved

Description

There are segments that contain unused state variables.

```
_previousOwner
_tOwned
```

Recommendation

Remove unused state variables.

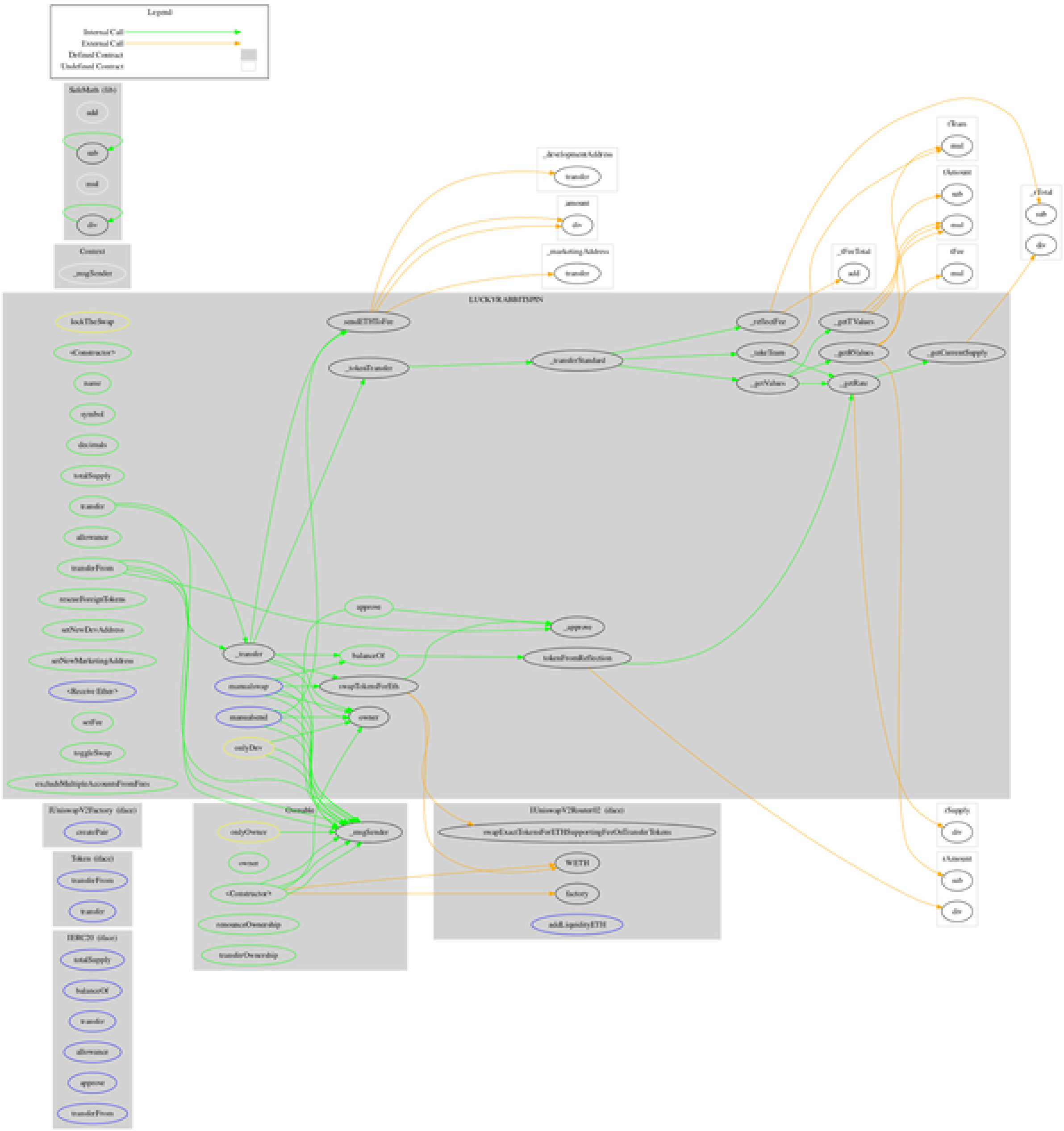
Contract Functions

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
Token	Interface			
	transferFrom	External	✓	-
	transfer	External	✓	-
IUniswapV2Factory	Interface			
	createPair	External	✓	-
IUniswapV2Router02	Interface			
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
	factory	External		-
	WETH	External		-
	addLiquidityETH	External	Payable	-
Context	Implementation			
	_msgSender	Internal		
SafeMath	Library			
	add	Internal		
	sub	Internal		

	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
LUCKYRABBITSPIN	Implementation	Context, IERC20, Ownable		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	tokenFromReflection	Private		
	_approve	Private	✓	
	_transfer	Private	✓	
	swapTokensForEth	Private	✓	lockTheSwap
	sendETHToFee	Private	✓	
	_tokenTransfer	Private	✓	
	rescueForeignTokens	Public	✓	onlyDev
	setNewDevAddress	Public	✓	onlyDev
	setNewMarketingAddress	Public	✓	onlyDev
	_transferStandard	Private	✓	
	_takeTeam	Private	✓	
	_reflectFee	Private	✓	

	<Receive Ether>	External	Payable	-
	_getValues	Private		
	_getTValues	Private		
	_getRValues	Private		
	_getRate	Private		
	_getCurrentSupply	Private		
	manualswap	External	✓	-
	manualsend	External	✓	-
	setFee	Public	✓	onlyDev
	toggleSwap	Public	✓	onlyDev
	excludeMultipleAccountsFromFees	Public	✓	onlyOwner

Contract Flow



Domain Info

Domain Name	luckyrabbit.games
Registry Domain ID	98ff80fe315e41e5bbac78aaa694114d-DONUTS
Creation Date	2022-12-05T07:00:49Z
Updated Date	2022-12-05T14:52:44Z
Registry Expiry Date	2023-12-05T07:00:49Z
Registrar WHOIS Server	whois.namecheap.com
Registrar URL	https://www.namecheap.com/
Registrar	NameCheap, Inc.
Registrar IANA ID	1068

The domain was created 2 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

There are some functions that can be abused by the owner like transferring the contract's tokens and transferring funds to the team's wallet. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats. There is also a limit of max 7% buy/sell fees.

Disclaimer

The information provided in this report does not constitute investment, financial or trading advice and you should not treat any of the document's content as such. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes nor may copies be delivered to any other person other than the Company without Cyberscope's prior written consent. This report is not nor should be considered an "endorsement" or "disapproval" of any particular project or team. This report is not nor should be regarded as an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Cyberscope to perform a security assessment. This document does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors' business, business model or legal compliance. This report should not be used in any way to make decisions around investment or involvement with any particular project. This report represents an extensive assessment process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk Cyberscope's position is that each company and individual are responsible for their own due diligence and continuous security Cyberscope's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies and in no way claims any guarantee of security or functionality of the technology we agree to analyze. The assessment services provided by Cyberscope are subject to dependencies and are under continuing development. You agree that your access and/or use including but not limited to any services reports and materials will be at your sole risk on an as-is where-is and as-available basis Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives false negatives and other unpredictable results. The services may access and depend upon multiple layers of third parties.

About Cyberscope

Cyberscope is a blockchain cybersecurity company that was founded with the vision to make web3.0 a safer place for investors and developers. Since its launch, it has worked with thousands of projects and is estimated to have secured tens of millions of investors’ funds.

Cyberscope is one of the leading smart contract audit firms in the crypto space and has built a high-profile network of clients and partners.

The Cyberscope team

https://www.cyberscope.io