



**AUDIT REPORT**

# **SecureWise**

**SMART CONTRACT AUDIT**



<https://github.com/securewise>



<https://t.me/securewise>



<https://securewise.info/>



# Table of Contents

**03**

Disclaimer

**04**

Overview

**05**

Quick Result

**06**

Auditing  
Approach and  
Methodologies

**07**

Automated  
Analysis

**11**

Inheritance  
Graph

**12**

Contract  
Summary

**13**

Manual Review



# Disclaimer

SecureWise provides the smart contract audit of solidity. Audit and report are for informational purposes only and not, nor should be considered, as an endorsement to engage with, invest in, participate, provide an incentive, or disapprove, criticise, discourage, or purport to provide an opinion on any particular project or team.

This audit report doesn't provide any warranty or guarantee regarding the nature of the technology analysed. These reports, in no way, provide investment advice, nor should be used as investment advice of any sort. Investors must always do their own research and manage their risk.

DISCLAIMER: By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and SecureWise and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) SecureWise owe no duty of care towards you or any other person, nor does SecureWise make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and SecureWise hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, SecureWise hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against SecureWise, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report.

# Overview

**Token Name:** Luck2Earn (**LUCK**)

**Methodology:** Automated Analysis, Manual Code Review

**Language:** Solidity

**Contract Address:** 0x1A63fe74d192E868fB484f25e86797EBACD705A7

**ContractLink:** <https://bscscan.com/address/0x1A63fe74d192E868fB484f25e86797EBACD705A7>

**Network:** Binance Smart Chain (BSC)

**Decimals:** 18

**Supply:** 100.000.000

**Website:** <https://www.lucktoearn.app/>

**Twitter:** -

**Telegram:** <https://t.me/lucktoearn>

**Report Date:** July 29, 2022

# Quick Result

SecureWise has applied the automated and manual analysis of Smart Contract and were reviewed for common contract vulnerabilities and centralized exploits

## Owner Privileges



The owner can set the max tx amount "0"



The owner can set fees up to 100%



The owner can exclude accounts from fees

**Luck2Earn LUCK** has succesfully **PASSED** the smart contract audit with **HIGH** and **LOW** severity issue

# Auditing Approach and Methodologies

SecureWise has performed starting with analyzing the code, issues, code quality, and libraries. Reviewed line-by-line by our team. Finding any potential issue like race conditions, transaction-ordering dependence, timestamp dependence, and denial of service attacks.

## Methodology

- Understanding the size, scope and functionality of your project's source code
- Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
- Testing and automated analysis of the Smart Contract to determine proper logic has been followed throughout the whole process
- Deploying the code on testnet using multiple live test
- Analyzing a program to determine the specific input that causes different parts of a program to execute its functions.
- Checking whether all the libraries used in the code are on the latest version.

## Goals

Smart Contract System is secure, resilient and working according to the specifications and without any vulnerabilities.



## Risk Classification































**High:** Exploits, vulnerabilities or errors that will certainly or probabilistically lead towards loss of funds, control, of the contract and its functions. Must be fixed as soon as possible.

**Medium:** Bugs or issues with that may be subject to exploit, though their impact is somewhat limited. Must be fixed as soon as possible.

**Low:** Effects are minimal in isolation and do not pose a significant danger to the project or its users. Issues under this classification are recommended to be fixed nonetheless.

# Automated Analysis

Symbol	Meaning
	Function can modify state
	Function is payable

































































<b>IERC20</b>	Interface			
L	totalSupply	External 		NO 
L	balanceOf	External 		NO 
L	transfer	External 		NO 
L	allowance	External 		NO 
L	approve	External 		NO 
L	transferFrom	External 		NO 
<b>SafeMath</b>	Library			
L	tryAdd	Internal 		
L	trySub	Internal 		
L	tryMul	Internal 		
L	tryDiv	Internal 		
L	tryMod	Internal 		
L	add	Internal 		
L	sub	Internal 		
L	mul	Internal 		
L	div	Internal 		
L	mod	Internal 		
L	sub	Internal 		
L	div	Internal 		
L	mod	Internal 		
<b>Context</b>	Implementation			
L	_msgSender	Internal 		
L	_msgData	Internal 		

# Automated Analysis

Address	Library			
L	isContract	Internal 🔒		
L	sendValue	Internal 🔒	●	
L	functionCall	Internal 🔒	●	
L	functionCall	Internal 🔒	●	
L	functionCallWithValue	Internal 🔒	●	
L	functionCallWithValue	Internal 🔒	●	
L	functionStaticCall	Internal 🔒		
L	functionStaticCall	Internal 🔒		
L	functionDelegateCall	Internal 🔒	●	
L	functionDelegateCall	Internal 🔒	●	
L	verifyCallResult	Internal 🔒		
Ownable	Implementation	Context		
L		Public 🚫	●	NO 🚫
L	owner	Public 🚫		NO 🚫
L	renounceOwnership	Public 🚫	●	onlyOwner
L	transferOwnership	Public 🚫	●	onlyOwner
L	_transferOwnership	Internal 🔒	●	



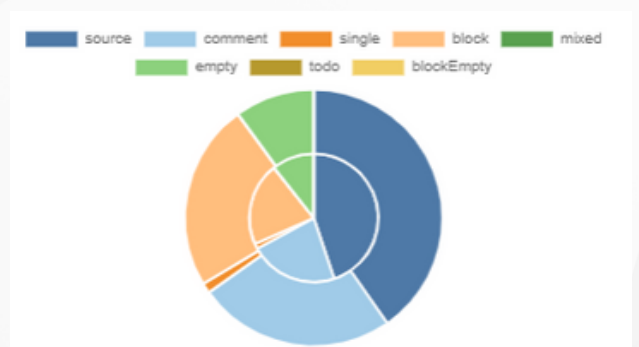
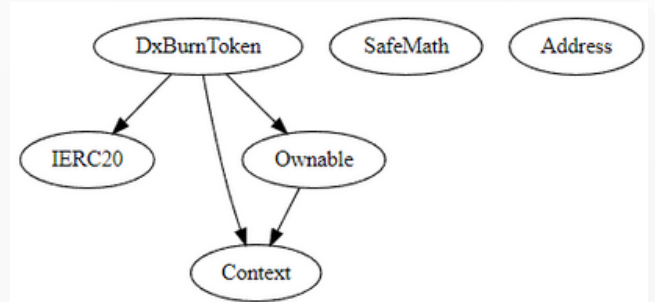
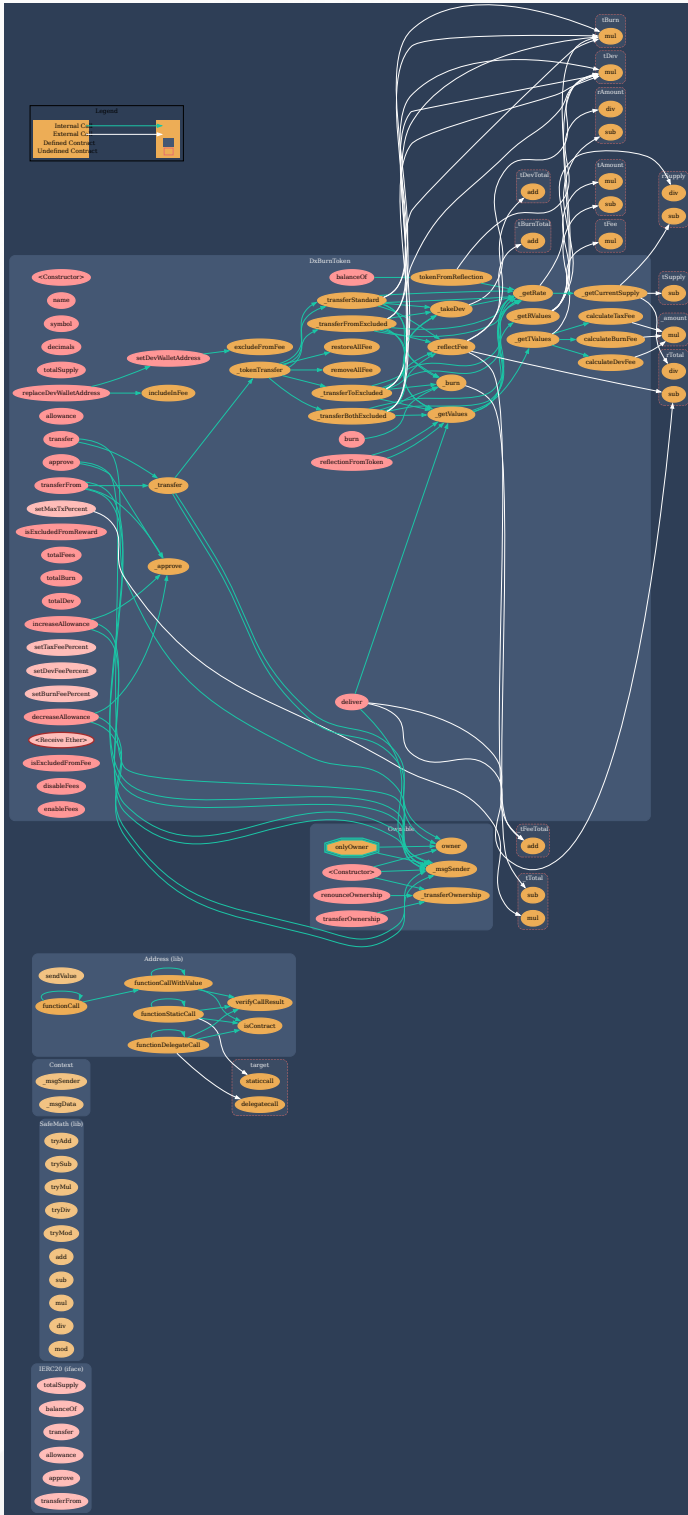
# Automated Analysis

DxBurnToken	Implementation	Context, IERC20, Ownable		
L		Public 		NO 
L	name	Public 		NO 
L	symbol	Public 		NO 
L	decimals	Public 		NO 
L	totalSupply	Public 		NO 
L	balanceOf	Public 		NO 
L	transfer	Public 		NO 
L	allowance	Public 		NO 
L	approve	Public 		NO 
L	transferFrom	Public 		NO 
L	increaseAllowance	Public 		NO 
L	decreaseAllowance	Public 		NO 
L	isExcludedFromReward	Public 		NO 
L	totalFees	Public 		NO 
L	totalBurn	Public 		NO 
L	totalDev	Public 		NO 
L	deliver	Public 		NO 
L	reflectionFromToken	Public 		NO 
L	tokenFromReflection	Public 		NO 
L	excludeFromFee	Public 		onlyOwner
L	includeInFee	Public 		onlyOwner
L	setDevWalletAddress	Public 		onlyOwner
L	replaceDevWalletAddress	Public 		onlyOwner
L	burn	Public 		NO 
L	setTaxFeePercent	External 		onlyOwner
L	setDevFeePercent	External 		onlyOwner
L	setBurnFeePercent	External 		onlyOwner
L	setMaxTxPercent	External 		onlyOwner

# Automated Analysis

L	_getValues	Private 🏠		
L	_getTValues	Private 🏠		
L	_getRValues	Private 🏠		
L	_getRate	Private 🏠		
L	_getCurrentSupply	Private 🏠		
L	_takeDev	Private 🏠	🔴	
L	calculateTaxFee	Private 🏠		
L	calculateDevFee	Private 🏠		
L	calculateBurnFee	Private 🏠		
L	removeAllFee	Private 🏠	🔴	
L	restoreAllFee	Private 🏠	🔴	
L	isExcludedFromFee	Public !		NO !
L	_burn	Private 🏠	🔴	
L	_approve	Private 🏠	🔴	
L	_transfer	Private 🏠	🔴	
L	_tokenTransfer	Private 🏠	🔴	
L	_transferStandard	Private 🏠	🔴	
L	_transferToExcluded	Private 🏠	🔴	
L	_transferFromExcluded	Private 🏠	🔴	
L	_transferBothExcluded	Private 🏠	🔴	
L	_reflectFee	Private 🏠	🔴	
L	disableFees	Public !	🔴	onlyOwner
L	enableFees	Public !	🔴	onlyOwner

# Inheritance Graph



# Contract Summary

Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
5	1	1098	974	563	363	383	
5	1	1098	974	563	363	383	

## Components

Contracts	Libraries	Interfaces	Abstract
1	2	1	2

## Exposed Functions

This section lists functions that are explicitly declared public or payable. Please note that getter methods for public stateVars are not included.

Public	Payable
40	1

External	Internal	Private	Pure	View
11	71	20	15	29

## StateVariables

Total	Public
34	15

## Capabilities

Solidity Versions observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
<input type="text" value="^0.8.7"/>	<input type="text" value="ABIEncoderV2"/>	<input type="text" value="yes"/>	<input type="text" value="yes (1 asm blocks)"/>	<input type="text"/>

Transfers ETH	Low-Level Calls	DelegateCall	Uses Hash Functions	ECREcover	New/Create/Create2
<input type="text"/>	<input type="text"/>	<input type="text" value="yes"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

TryCatch	Unchecked
<input type="text"/>	<input type="text" value="yes"/>

# Manual Review

## The owner can set the max tx amount "0"

```
847     function setMaxTxPercent(uint256 maxTxPercent) external onlyOwner() {
848         require(maxTxPercent >= minMxTxPercentage && maxTxPercent <=100,"maxTxPercent out of range");
849         _maxTxAmount = _tTotal.mul(maxTxPercent).div(
850             10**2
851         );
852     }
```

### Recommendation

These functions should be provided arbitrary limits, e.g., put a **require** check that allows maximum limit etc. if **set 0** these cause pause the trading.

## The owner can set fees up to 100%

```
829     function setTaxFeePercent(uint256 taxFee) external onlyOwner() {
830         require(taxFee >= 0 && taxFee <=maxTaxFee,"taxFee out of range");
831         _taxFee = taxFee;
832     }
833
834     function setDevFeePercent(uint256 devFee) external onlyOwner() {
835         require(devFee >= 0 && devFee <=maxDevFee,"teamFee out of range");
836         _devFee = devFee;
837     }
838
839     function setBurnFeePercent(uint256 burnFee) external onlyOwner() {
840         require(burnFee >= 0 && burnFee <=maxBurnFee,"teamFee out of range");
841         _burnFee = burnFee;
842     }
843 }
```

### Recommendation

These functions should be provided arbitrary limits, e.g., put a **require** check that allows maximum limit etc.

# Manual Review

## The owner can exclude accounts from fees

```
795     function excludeFromFee(address account) public onlyOwner {
796         require(!_isExcludedFromFee[account], "Account is already excluded");
797         _isExcludedFromFee[account] = true;
798     }
799
800     function includeInFee(address account) public onlyOwner {
801         require(_isExcludedFromFee[account], "Account is already included");
802         _isExcludedFromFee[account] = false;
803     }
804
```

### Recommendation

Authorizing privileged roles to exclude accounts from fees. These cause affect decentralization

## AUDIT REPORT

# SecureWise

## SMART CONTRACT AUDIT

 <https://github.com/securewise>  
 <https://t.me/securewise>  
 <https://securewise.info/>

