



RUGFREECOINS



# Cake Bot Token

RugfreeCoins Verified on October 26th, 2023

# Overview

- ✓ No mint function found, the owner cannot mint tokens after initial deployment.
- ✓ The owner can't set a max transaction limit
- ✓ The owner can't pause trading once it's enabled
- ✗ The owner must enable trade for the holders, if trading remains disabled, no one would be able to buy and sell.
- ✓ The owner can't change fees over 25%.
- ✓ The owner can't blacklist wallets.
- ✓ The owner can't set a max wallet limit
- ✓ The owner can't claim the contract's balance of its own token.

## ! HIGH SEVERITY ISSUES

The owner must enable trade for the holders, if trading remains disabled, no one would be able to buy and sell.

```
function enableTrading() external onlyOwner {  
    require(!tradingEnabled, "Trading already enabled.");  
    tradingEnabled = true;  
}
```

# Contents

Overview.....	2
Contents.....	3
Audit details.....	4
Disclaimer.....	5
Background.....	6
Tokenomics.....	7
Target market and the concept.....	8
Potential to grow with score points.....	9
Contract details.....	10
Contract code function details.....	11
Contract description table.....	12
Inheritance Hierarchy.....	17
Security issue checking status.....	18
Owner privileges.....	19
Audit conclusion.....	22

## Audit details



**Audited project**  
Cake Bot Token



**Contract Address**  
0x1F4e01Bd38cC9f2aEF4187Af505cf31535af28cF



**Client contact**  
Cake Bot Token Team



**Blockchain**  
Binance Smart chain



**Project website**  
<https://cakebot.net>

# Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the disclaimer below – **please make sure to read it in full.**

## ❗ 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 **RugfreeCoins and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (RugfreeCoins) owe no duty of care towards you or any other person**, nor does RugfreeCoins 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 RugfreeCoins 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, RugfreeCoins hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against RugfreeCoins, 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.

The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

# Background

**RugfreeCoins** was commissioned by the **Cake Bot Token Team** to perform an audit of the smart contract.

<https://bscscan.com/token/0x1f4e01bd38cc9f2aef4187af505cf31535af28cf>

This audit focuses on verifying that the smart contract is secure, resilient, and working according to the specifications.

The information in this report should be used to understand the risk exposure of the smart contract, project feasibility, and long-term sustainability, and as a guide to improving the smart contract's security posture by remediating the identified issues.

# Tokenomics

## ▲ 5% tax when buying & selling

3% trade is distributed among holders as rewards in tokens.

1% of trade goes to the marketing wallet in BNB

1% of trade goes to the Liquidity Pool












---

# Target market and the concept

- ▶ Anyone who's interested in the Crypto space with long-term investment plans.
- ▶ Anyone who's ready to earn a passive income by holding tokens.
- ▶ Anyone who's interested in trading tokens.
- ▶ Anyone who's interested in taking part in the Cake Bot token ecosystem.
- ▶ Anyone who's interested in taking part in the future plans of Cake Bot Token.
- ▶ Anyone who's interested in making financial transactions with any other party using Cake Bot Token as the currency.



## Potential to grow with score points

 Project efficiency	9 / 10
 Project uniqueness	9 / 10
 Information quality	8 / 10
 Service quality	8 / 10
 System quality	8 / 10
 Impact on the community	9 / 10
 Impact on the business	9 / 10
 Preparing for the future	9 / 10
 Smart contract security	9 / 10
 Smart contract functionality assessment	9 / 10
 <b>Total Score</b>	<b>8.7 / 10</b>

# Contract details

Token contract details for 26th of October 2023

Contract name	<b>CakeBot</b>
Contract address	<b>0x1F4e01Bd38cC9f2aEF4187Af505cf31535af28cF</b>
Token supply	<b>10</b>
Token ticker	<b>CAKEBOT</b>
Decimals	<b>9</b>
Token holders	<b>1</b>
Transaction count	<b>1</b>
Contract deployer address	<b>0x1Ae9fB29af80B36caB2f4822EF25ae741013EE2F</b>
Contract's current owner address	<b>0x1Ae9fB29af80B36caB2f4822EF25ae741013EE2F</b>
Marketing wallet	<b>0xe47169B17503C2C7fF60876A83F0F754E63fdC7f</b>





































# Contract code function details

















Nº	Category	Item	Result
1	Coding conventions	ERC20 Token standards	PASS ▾
		Compile errors	PASS ▾
		Compiler version security	PASS ▾
		Visibility specifiers	PASS ▾
		Gas consumption	PASS ▾
		SafeMath features	PASS ▾
		Fallback usage	PASS ▾
		tx.origin usage	PASS ▾
		Deprecated items	PASS ▾
		Redundant code	PASS ▾
2	Function call audit	Overriding variables	PASS ▾
		Authorization of function call	PASS ▾
		Low level function (call/delegate call) security	PASS ▾
		Returned value security	PASS ▾
3	Business security & centralisation	Self destruct function security	PASS ▾
		Access control of owners	HIGH ▾
		Business logics	LOW ▾
4	Integer overflow/underflow	Business implementation	PASS ▾
5	Reentrancy		PASS ▾
6	Exceptional reachable state		PASS ▾
7	Transaction ordering dependence		PASS ▾
8	Block properties dependence		PASS ▾
9	Pseudo random number generator (PRNG)		PASS ▾
10	DoS (Denial of Service)		PASS ▾
11	Token vesting implementation		PASS ▾
12	Fake deposit		PASS ▾
13	Event security		PASS ▾

# Contract description table

The below table represents the summary of the contracts and methods in the token contract. We scanned the whole contract and listed down all the Interfaces, functions, and implementations with their visibility and mutability.

Contract	Type	Bases		
L	Function Name	Visibility	Mutability	Modifiers
<b>IERC20</b>	<b>Interface</b>			
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>Context</b>	<b>Implementation</b>			
L	_msgSender	Internal 🔒		
L	_msgData	Internal 🔒		
<b>Ownable</b>	<b>Implementation</b>	Context		
L		Public !	⛔	NO !
L	owner	Public !		NO !
L	renounceOwnership	Public !	⛔	onlyOwner
L	transferOwnership	Public !	⛔	onlyOwner
L	_setOwner	Private 🔒	⛔	
<b>SafeMath</b>	<b>Library</b>			
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>Address</b>	<b>Library</b>			
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 		
<b>IUniswapV2 Router01</b>	<b>Interface</b>			
L	factory	External 		NO 
L	WETH	External 		NO 
L	addLiquidity	External 		NO 

L	addLiquidityETH	External !		NO !
L	removeLiquidity	External !		NO !
L	removeLiquidityETH	External !		NO !
L	removeLiquidityWithPermit	External !		NO !
L	removeLiquidityETHWithPermit	External !		NO !
L	swapExactTokensForTokens	External !		NO !
L	swapTokensForExactTokens	External !		NO !
L	swapExactETHForTokens	External !		NO !
L	swapTokensForExactETH	External !		NO !
L	swapExactTokensForETH	External !		NO !
L	swapETHForExactTokens	External !		NO !
L	quote	External !		NO !
L	getAmountOut	External !		NO !
L	getAmountIn	External !		NO !
L	getAmountsOut	External !		NO !
L	getAmountsIn	External !		NO !
<b>IUniswapV2 Router02</b>	<b>Interface</b>	<b>IUniswapV2 Router01</b>		
L	removeLiquidityETHSupportingFeeOnTransferTokens	External !		NO !
L	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External !		NO !
L	swapExactTokensForTokensSupportingFeeOnTransferTokens	External !		NO !
L	swapExactETHForTokensSupportingFeeOnTransferTokens	External !		NO !
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External !		NO !
<b>IUniswapV2 Factory</b>	<b>Interface</b>			
L	feeTo	External !		NO !

L	feeToSetter	External !		NO !
L	getPair	External !		NO !
L	allPairs	External !		NO !
L	allPairsLength	External !		NO !
L	createPair	External !		NO !
L	setFeeTo	External !		NO !
L	setFeeToSetter	External !		NO !
<b>CakeBot</b>	<b>Implementation</b>	<b>IERC20, Ownable</b>		
L		Public !		NO !
L	name	Public !		NO !
L	symbol	Public !		NO !
L	decimals	Public !		NO !
L	totalSupply	Public !		NO !
L	balanceOf	Public !		NO !
L	enableTrading	External !		onlyOwner
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	deliver	Public !		NO !
L	reflectionFromToken	Public !		NO !
L	tokenFromReflection	Public !		NO !
L	excludeFromReward	Public !		onlyOwner
L	includeInReward	External !		onlyOwner

L	_transferBothExcluded	Private 🔒	🛑	
L	excludeFromFee	Public !	🛑	onlyOwner
L	setTaxFeePercent	External !	🛑	onlyOwner
L	setLiquidityFeePercent	External !	🛑	onlyOwner
L	setMarketingFeePercent	External !	🛑	onlyOwner
L	setSwapBackSettings	External !	🛑	onlyOwner
L		External !	📊	NO !
L	_reflectFee	Private 🔒	🛑	
L	_getValues	Private 🔒		
L	_getTValues	Private 🔒		
L	_getRValues	Private 🔒		
L	_getRate	Private 🔒		
L	_getCurrentSupply	Private 🔒		
L	_takeLiquidity	Private 🔒	🛑	
L	_takeMarketingFee	Private 🔒	🛑	
L	calculateTaxFee	Private 🔒		
L	calculateLiquidityFee	Private 🔒		
L	calculateMarketingFee	Private 🔒		
L	removeAllFee	Private 🔒	🛑	
L	restoreAllFee	Private 🔒	🛑	
L	isExcludedFromFee	Public !		NO !
L	_approve	Private 🔒	🛑	
L	_transfer	Private 🔒	🛑	
L	swapAndLiquify	Private 🔒	🛑	lockThe Swap
L	swapTokensForEth	Private 🔒	🛑	
L	addLiquidity	Private 🔒	🛑	
L	_tokenTransfer	Private 🔒	🛑	
L	_transferStandard	Private 🔒	🛑	

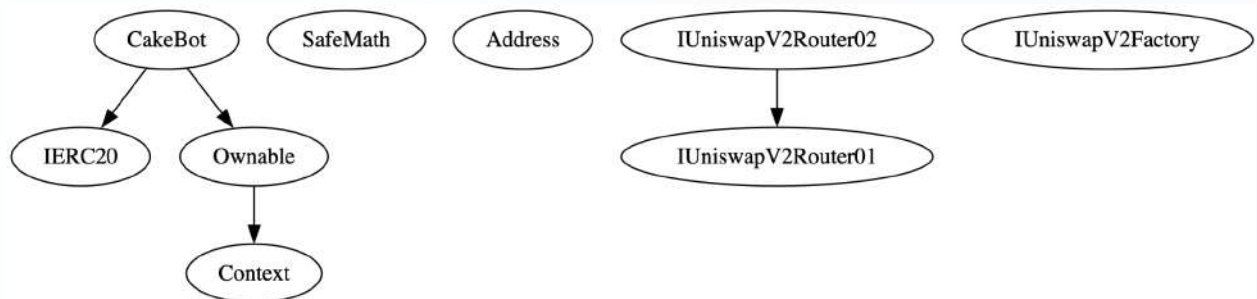


L	_transferToExcluded	Private 🔒	🛑	
L	_transferFromExcluded	Private 🔒	🛑	

### Legend

Symbol	Meaning
🛑	Function can modify state
💰	Function is payable

## Inheritance Hierarchy



# Security issue checking status

## ❖ High severity issues

The owner must enable trade for the holders, if trading remains disabled, no one would be able to buy and sell.

```
function enableTrading() external onlyOwner {  
    require(!tradingEnabled, "Trading already enabled.");  
    tradingEnabled = true;  
}
```

## ❖ Medium severity issues

No medium severity issues found

## ❖ Low severity issues

The owner can exclude wallets from fees and, once excluded, can not include them again.

```
function excludeFromFee(address account) public onlyOwner {  
    _isExcludedFromFee[account] = true;  
}
```

# Owner privileges

- ❖ The owner can enable trading, but once enabled can not disable it again

```
function enableTrading() external onlyOwner {
    require(!tradingEnabled, "Trading already enabled.");
    tradingEnabled = true;
}
```

- ❖ The owner can include/exclude wallets from rewards

```
function excludeFromReward(address account) public onlyOwner {
    // require(account != 0x7a250d5630B4cF539739dF2C5dAcb4c659F2488D, 'We can
    not exclude Uniswap router. ');
    require(!_isExcluded[account], "Account is already excluded");
    if (_rOwned[account] > 0) {
        _tOwned[account] = tokenFromReflection(_rOwned[account]);
    }
    _isExcluded[account] = true;
    _excluded.push(account);
}

function includeInReward(address account) external onlyOwner {
    require(_isExcluded[account], "Account is already excluded");
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (_excluded[i] == account) {
            _excluded[i] = _excluded[_excluded.length - 1];
            _tOwned[account] = 0;
            _isExcluded[account] = false;
            _excluded.pop();
            break;
        }
    }
}
```

- ❖ The owner can exclude wallets from fees, once excluded can not include them again

```
function excludeFromFee(address account) public onlyOwner {  
    _isExcludedFromFee[account] = true;  
}
```

- ❖ The owner can change all fees, with total fees maximum of up-to 25%

```
function setTaxFeePercent(uint256 taxFeeBps) external onlyOwner {  
    _taxFee = taxFeeBps;  
    require(  
        _taxFee + _liquidityFee + _marketingFee <= MAX_FEE,  
        "Total fee is over 25%"  
    );  
}  
  
function setLiquidityFeePercent(  
    uint256 liquidityFeeBps  
) external onlyOwner {  
    _liquidityFee = liquidityFeeBps;  
    require(  
        _taxFee + _liquidityFee + _marketingFee <= MAX_FEE,  
        "Total fee is over 25%"  
    );  
}  
  
function setMarketingFeePercent(  
    uint256 marketingFeeBps  
) external onlyOwner {  
    _marketingFee = marketingFeeBps;  
    require(  
        _taxFee + _liquidityFee + _marketingFee <= MAX_FEE,  
        "Total fee is over 25%"  
    );  
}
```

- ❖ The owner can change the swapback limit minimum up-to 0.05%

```
function setSwapBackSettings(uint256 _amount) external onlyOwner {
    require(
        _amount >= totalSupply().mul(5).div(10 ** 4),
        "Swapback amount should be at least 0.05% of total supply"
    );
    numTokensSellToAddToLiquidity = _amount;
    emit SwapAndLiquifyAmountUpdated(_amount);
}
```

## Audit conclusion

RugFreeCoins team has performed in-depth testing, line-by-line manual code review, and automated audit of the smart contract. The smart contract was analyzed mainly for common smart contract vulnerabilities, exploits, manipulations, and hacks. According to the smart contract audit.

Smart contract functional Status:	PASS ▾
Smart contract security Status:	HIGH ISSUES ▾
Number of risk issues:	2
Solidity code functional issue level:	PASS ▾
Number of owner privileges:	5
Centralization risk correlated to the active owner:	HIGH ▾
Smart contract active ownership:	ACTIVE ▾