



RugFreeCoins Audit



Its CORN Token Smart Contract Security Audit

June 18th ,2023

Contents

Audit details	1
Disclaimer	2
Overview	3
Background	4
Target market and the concept	6
Contract details	7
Contract code function details	8
Contract description table	10
Security issue checking status	23
Owner privileges	25
Audit conclusion	32

Audit details



Audited project
ITs CORN Token



Contract Address
0xa66b9baCA9681C154890AD703dc5e2766d4ec2C1



Client contact
ITs CORN Token Team



Blockchain
Binance smart chain



Project website
<https://itscorn token.com/>

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.

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 enable or pause trading.
- ✓ The owner can't change fees more than 25%.
- ✓ The owner can't blacklist wallets.
- ✓ The owner can't set a max wallet limit of less than 0.1%
- ✓ The owner can't claim the contract's balance of its own token.

Background

Rugfreecoins was commissioned by the ITs CORN Token Team to perform an audit of the smart contract.

<https://bscscan.com/token/0xa66b9baCA9681C154890AD703dc5e2766d4ec2C1>

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

7% tax when buying and selling

- 1% of trade goes to the marketing wallet in BNB
- 1% of trade goes to the Dev wallet in BNB
- 1% of trade goes to the burn wallet
- 2% of trade goes to the liquidity pool
- 2% trade distributes among holders as rewards in BNBs.

Target market and the concept

Target market

- 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 ITs CORN token ecosystem.
- Anyone who's interested in taking part in the future plans of ITs CORN Token.
- Anyone who's interested in making financial transactions with any other party using ITs CORN Token as the currency.

Contract details

Token contract details for 18th of June 2023

Contract name	ITs CORN
Contract address	0xa66b9baCA9681C154890AD703dc5e2766d4ec2C1
Token supply	100,000,000,000
Token ticker	ICN
Decimals	18
Token holders	1
Transaction count	1
Contract deployer address	0xc3B8Aee6f6EB1E6f6d2D7f52508f620795cFE3F9
Contract's current owner address	0xc3b8aee6f6eb1e6f6d2d7f52508f620795cfe3f9
Dividend Tracker	0x44908656f664f9688ae9353310525626865ca250
Marketing address	0x9a1c597291dc17ce9ffdc4c10e849b7c927a2d9d






Contract code function details








No	Category	Item	Result
1	Coding conventions	BRC20 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
		Overriding variables	pass
2	Function call audit	Authorization of function call	pass
		Low level function (call/delegate call) security	pass
		Returned value security	pass
		Selfdestruct function security	pass
3	Business security & centralization	Access control of owners	pass
		Business logics	pass
		Business implementations	pass
4	Integer overflow/underflow		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
Context	Implementation			
L	_msgSender	Internal 		
L	_msgData	Internal 		
IUniswapV2Pair	Interface			
L	name	External !		NO !
L	symbol	External !		NO !
L	decimals	External !		NO !
L	totalSupply	External !		NO !
L	balanceOf	External !		NO !
L	allowance	External !		NO !
L	approve	External !		NO !
L	transfer	External !		NO !
L	transferFrom	External !		NO !
L	DOMAIN_SEPARATOR	External !		NO !
L	PERMIT_TYPEHASH	External !		NO !











L	nonces	External !		NO !
L	permit	External !		NO !
L	MINIMUM_LIQUIDITY	External !		NO !
L	factory	External !		NO !
L	token0	External !		NO !
L	token1	External !		NO !
L	getReserves	External !		NO !
L	price0CumulativeLast	External !		NO !
L	price1CumulativeLast	External !		NO !
L	kLast	External !		NO !
L	mint	External !		NO !
L	burn	External !		NO !
L	swap	External !		NO !
L	skim	External !		NO !
L	sync	External !		NO !
L	initialize	External !		NO !
IUniswapV2 Factory	Interface			
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 !
IERC20	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 !
IERC20Metadata	Interface	IERC20		
L	name	External !		NO !
L	symbol	External !		NO !
L	decimals	External !		NO !
ERC20	Implementation	Context, IERC20, IERC20 Metadata		
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	_transfer	Internal 🔒	🔴	
L	_mint	Internal 🔒	🔴	
L	_burn	Internal 🔒	🔴	
L	_approve	Internal 🔒	🔴	
L	_beforeTokenTransfer	Internal 🔒	🔴	
DividendPaying Token Optional Interface	Interface			
L	withdrawableDividendOf	External !		NO !
L	withdrawnDividendOf	External !		NO !

L	accumulativeDividendOf	External !		NO !
DividendPayingTokenInterface	Interface			
L	dividendOf	External !		NO !
L	distributeDividends	External !	🔒	NO !
L	withdrawDividend	External !	🔴	NO !
SafeMath	Library			
L	add	Internal 🔒		
L	sub	Internal 🔒		
L	sub	Internal 🔒		
L	mul	Internal 🔒		
L	div	Internal 🔒		
L	div	Internal 🔒		
L	mod	Internal 🔒		
L	mod	Internal 🔒		
Ownable	Implementation	Context		
L		Public !	🔴	NO !
L	owner	Public !		NO !
L	renounceOwnership	Public !	🔴	onlyOwner
L	transferOwnership	Public !	🔴	onlyOwner

SafeMathInt	Library			
L	mul	Internal 🔒		
L	div	Internal 🔒		
L	sub	Internal 🔒		
L	add	Internal 🔒		
L	abs	Internal 🔒		
L	toUint256Safe	Internal 🔒		
SafeMathUint	Library			
L	toInt256Safe	Internal 🔒		
IUniswapV2 Router01	Interface			
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 !
IUniswapV2 Router02	Interface	IUniswapV2 Router01		
L	removeLiquidityETHSupportingFeeOnTransferTokens	External !		NO !
L	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External !		NO !
L	swapExactTokensForTokensSupportingFeeOnTransferTokens	External !		NO !
L	swapExactETHForTokensSupportingFeeOnTransferTokens	External !		NO !
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External !		NO !
DividendPaying Token	Implementation	ERC20, DividendPaying Token Interface, DividendPayingToken Optional Interface		

L		Public !		ERC20
L		External !		NO !
L	distributeDividends	Public !		NO !
L	withdrawDividend	Public !		NO !
L	_withdrawDividendOfUser	Internal 		
L	dividendOf	Public !		NO !
L	withdrawableDividendOf	Public !		NO !
L	withdrawnDividendOf	Public !		NO !
L	accumulativeDividendOf	Public !		NO !
L	_transfer	Internal 		
L	_mint	Internal 		
L	_burn	Internal 		
L	_setBalance	Internal 		
ITsCORN	Implementation	ERC20, Ownable		
L		Public !		ERC20
L	decimals	Public !		NO !
L		External !		NO !
L	updateStakingAmounts	Public !		onlyOwner
L	setPresaleWallet	External !		onlyOwner
L	setExcludeFees	Public !		onlyOwner

L	setExcludeDividends	Public !		onlyOwner
L	setIncludeDividends	Public !		onlyOwner
L	setLimitsInEffect	External !		onlyOwner
L	setcooldowntimer	External !		onlyOwner
L	setMaxWallet	External !		onlyOwner
L	enableStaking	Public !		onlyOwner
L	stake	Public !		NO !
L	setSwapTriggerAmount	Public !		onlyOwner
L	enableSwapAndLiquify	Public !		onlyOwner
L	setAutomatedMarketMakerPair	Public !		onlyOwner
L	setAllowCustomTokens	Public !		onlyOwner
L	setAllowAutoReinvest	Public !		onlyOwner
L	_setAutomatedMarketMakerPair	Private 🗝️		
L	updateGasForProcessing	Public !		onlyOwner
L	transferAdmin	Public !		onlyOwner
L	updateTransferFee	Public !		onlyOwner
L	updateFees	Public !		onlyOwner
L	getStakingInfo	External !		NO !
L	getTotalDividendsDistributed	External !		NO !
L	isExcludedFromFees	Public !		NO !
L	withdrawableDividendOf	Public !		NO !



L	dividendTokenBalanceOf	Public !		NO !
L	getAccountDividendsInfo	External !		NO !
L	getAccountDividendsInfoAtIndex	External !		NO !
L	processDividendTracker	External !	🔴	NO !
L	claim	External !	🔴	NO !
L	getLastProcessedIndex	External !		NO !
L	getNumberOfDividendTokenHolders	External !		NO !
L	setAutoClaim	External !	🔴	NO !
L	setReinvest	External !	🔴	NO !
L	isExcludedFromAutoClaim	External !		NO !
L	isReinvest	External !		NO !
L	_transfer	Internal 🗝️	🔴	
L	getStakingBalance	Private 🗝️		
L	swapAndLiquify	Private 🗝️	🔴	
L	swapTokensForEth	Private 🗝️	🔴	
L	updatePayoutToken	Public !	🔴	onlyOwner
L	getPayoutToken	Public !		NO !
L	setMinimumTokenBalanceForAutoDividends	Public !	🔴	onlyOwner
L	setMinimumTokenBalanceForDividends	Public !	🔴	onlyOwner
L	addLiquidity	Private 🗝️	🔴	
L	forceSwapAndSendDividends	Public !	🔴	onlyOwner

L	swapAndSendDividends	Private 🗝️	🔴	
L	multiSend	Public !	🔴	onlyOwner
L	airdropToWallets	External !	🔴	onlyOwner
ITsCORN DividendTracker	Implementation	Dividend Paying Token, Ownable		
L		Public !	🔴	DividendPa yingToken
L	decimals	Public !		NO !
L	name	Public !		NO !
L	symbol	Public !		NO !
L	_transfer	Internal 🗝️		
L	withdrawDividend	Public !		NO !
L	isExcludedFromAutoClaim	External !		onlyOwner
L	isReinvest	External !		onlyOwner
L	setAllowCustomTokens	External !	🔴	onlyOwner
L	setAllowAutoReinvest	External !	🔴	onlyOwner
L	excludeFromDividends	External !	🔴	onlyOwner
L	includeFromDividends	External !	🔴	onlyOwner
L	setAutoClaim	External !	🔴	onlyOwner
L	setReinvest	External !	🔴	onlyOwner
L	setMinimumTokenBalanceForAuto Dividends	External !	🔴	onlyOwner

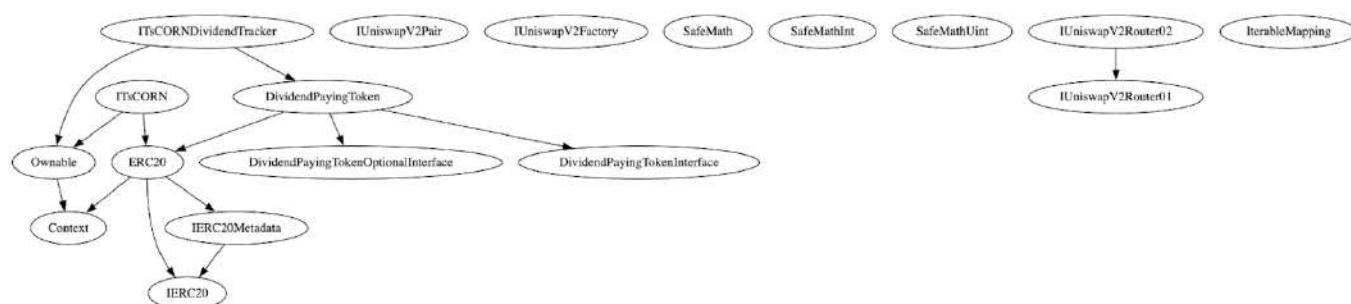
L	setMinimumTokenBalanceForDividends	External !		onlyOwner
L	setDividendsPaused	External !		onlyOwner
L	getLastProcessedIndex	External !		NO !
L	getNumberOfTokenHolders	External !		NO !
L	getAccount	Public !		NO !
L	getAccountAtIndex	Public !		NO !
L	setBalance	External !		onlyOwner
L	process	Public !		NO !
L	processAccount	Public !		onlyOwner
L	updateUniswapV2Router	Public !		onlyOwner
L	updatePayoutToken	Public !		onlyOwner
L	getPayoutToken	Public !		NO !
L	_reinvestDividendOfUser	Private 		
L	_withdrawDividendOfUser	Internal 		
IterableMapping	Library			
L	get	Internal 		
L	getIndexOfKey	Internal 		
L	getKeyAtIndex	Internal 		
L	size	Internal 		
L	set	Internal 		

L	remove	Internal 		
---	--------	--	---	--

Legend

Symbol	Meaning
	Function can modify state
	Function is payable

Inheritance Hierarchy



Security issue checking status

❖ High severity issues

Owner can set buy and sell fees up to 100% (If buy and sell fees are set to 100% it's not tradable)

Informed & Fixed

```
function updateFees(
    uint256 deadBuy,
    uint256 deadSell,
    uint256 marketingBuy,
    uint256 marketingSell,
    uint256 liquidityBuy,
    uint256 liquiditySell,
    uint256 RewardsBuy,
    uint256 RewardsSell,
    uint256 devBuy,
    uint256 devSell
) public onlyOwner {
    buyDeadFees = deadBuy;
    buyMarketingFees = marketingBuy;
    buyLiquidityFee = liquidityBuy;
    buyRewardsFee = RewardsBuy;
    sellDeadFees = deadSell;
    sellMarketingFees = marketingSell;
    sellLiquidityFee = liquiditySell;
    sellRewardsFee = RewardsSell;
    buyDevFee = devBuy;
    sellDevFee = devSell;

    totalSellFees = sellRewardsFee
        .add(sellLiquidityFee)
        .add(sellMarketingFees)
        .add(sellDevFee);

    totalBuyFees = buyRewardsFee
        .add(buyLiquidityFee)
        .add(buyMarketingFees)
        .add(buyDevFee);

    require(
        totalSellFees <= 100 && totalBuyFees <= 100,
        "total fees cannot exceed 15% sell or buy"
    );
}
```

Owner can disable dividend any time

Informed & Fixed

```
function setDividendsPaused(bool value) external onlyOwner {  
    dividendTracker.setDividendsPaused(value);  
}
```

❖ **Medium severity issues**

No medium severity issues found

❖ **Low severity issues**

No low severity issues found

❖ **Centralization Risk**

Auto LP is not going to an unreachable address

Informed & Fixed

```
function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {  
    // approve token transfer to cover all possible scenarios  
    _approve(address(this), address(uniswapV2Router), tokenAmount);  
  
    // add the liquidity  
    uniswapV2Router.addLiquidityETH{value: ethAmount}(  
        address(this),  
        tokenAmount,  
        0, // slippage is unavoidable  
        0, // slippage is unavoidable  
        owner(),  
        block.timestamp  
    );  
}
```


Owner privileges

- ❖ Owner can update staking bonus amount to each durations

```
function updateStakingAmounts(  
    uint256 duration,  
    uint256 bonus  
) public onlyOwner {  
    require(stakingAmounts[duration] != bonus);  
    require(bonus <= 100, "Staking bonus can't exceed 100");  
    stakingAmounts[duration] = bonus;  
    emit UpdateStakingAmounts(duration, bonus);  
}
```

- ❖ Owner can whitelist pre-sale wallet (exclude from fees, dividend and able to transfer before enabling trading)

```
// use for pre sale wallet, adds all exclusions to it  
function setPresaleWallet(address wallet) external onlyOwner {  
    canTransferBeforeTradingIsEnabled[wallet] = true;  
    _isExcludedFromFees[wallet] = true;  
    dividendTracker.excludeFromDividends(wallet);  
    emit SetPreSaleWallet(wallet);  
}
```

- ❖ Owner can include/exclude wallets from fees

```
// exclude a wallet from fees  
function setExcludeFees(address account, bool excluded) public onlyOwner {  
    _isExcludedFromFees[account] = excluded;  
    emit ExcludeFromFees(account, excluded);  
}
```

```
// exclude from dividends (rounds)
```

- ❖ Owner can exclude wallets from dividend

```
// exclude from dividends (rewards)
function setExcludeDividends(address account) public onlyOwner {
    dividendTracker.excludeFromDividends(account);
}
```

- ❖ Owner can include wallets from dividend

```
// include in dividends
function setIncludeDividends(address account) public onlyOwner {
    dividendTracker.includeFromDividends(account);
    dividendTracker.setBalance(account, getStakingBalance(account));
}
```

- ❖ Owner can enable/disable time limit and max wallet limit

```
// turn limits on and off
function setLimitsInEffect(bool value) external onlyOwner {
    limitsInEffect = value;
}
```

- ❖ Owner can set a max gas limit when buying a token, minimum 5 Gwei (if transaction take more than that limit, tx will fail)

```
// set max GWEI
function setGasPriceLimit(uint256 GWEI) external onlyOwner {
    require(GWEI >= 5, "can never be set below 5");
    gasPriceLimit = GWEI * 1 gwei;
}
```

- ❖ Owner can set cooldown timer maximum up to 15-sec

```
// set cooldown timer, can only be between 0 and 300 seconds (5 mins max)
function setCooldownTimer(uint256 value) external onlyOwner {
    require(value <= 15, "cooldown timer cannot exceed 5 minutes");
    cooldownTimer = value;
}
```

- ❖ Owner can set max wallet limit minimum up to 0.05%

```
// set max wallet, can not be lower than 0.05% of supply
function setMaxWallet(uint256 value) external onlyOwner {
    value = value * (10 ** 18);
    require(
        value >= _totalSupply / 2000,
        "max wallet cannot be set to less than 0.05%"
    );
    maxWallet = value;
}
```

- ❖ Owner can enable/disable staking

```
function enableStaking(bool enable) public onlyOwner {
    require(stakingEnabled != enable);
    stakingEnabled = enable;
    emit EnableStaking(enable);
}
```

- ❖ Owner can change swap point

```
// rewards threshold
function setSwapTriggerAmount(uint256 amount) public onlyOwner {
    swapTokensAtAmount = amount * (10 ** 18);
}
```

- ❖ Owner can enable/disable adding LP

```
function enableSwapAndLiquify(bool enabled) public onlyOwner {  
    require(swapAndLiquifyEnabled != enabled);  
    swapAndLiquifyEnabled = enabled;  
    emit EnableSwapAndLiquify(enabled);  
}
```

- ❖ Owner can add/remove LP pairs

```
function setAutomatedMarketMakerPair(  
    address pair,  
    bool value  
) public onlyOwner {  
    _setAutomatedMarketMakerPair(pair, value);  
}
```

- ❖ Owner allow/disallow to send custom tokens as dividend

```
function setAllowCustomTokens(bool allow) public onlyOwner {  
    dividendTracker.setAllowCustomTokens(allow);  
}
```

- ❖ Owner can enable/disable auto reinvest

```
function setAllowAutoReinvest(bool allow) public onlyOwner {  
    dividendTracker.setAllowAutoReinvest(allow);  
}
```

- ❖ Owner can transfer ownership

```
function transferAdmin(address newOwner) public onlyOwner {  
    dividendTracker.excludeFromDividends(newOwner);  
    _isExcludedFromFees[newOwner] = true;  
    transferOwnership(newOwner);  
}
```

- ❖ Owner can change transfer fees maximum upto 15%

```
function updateTransferFee(uint256 newTransferFee) public onlyOwner {  
    require(newTransferFee <= 15, "transfer fee cannot exceed 15%");  
    transferFee = newTransferFee;  
    emit UpdateTransferFee(transferFee);  
}
```

- ❖ Owner can pause/resume dividend any time

```
function setDividendsPaused(bool value) external onlyOwner {  
    dividendTracker.setDividendsPaused(value);  
}
```

- ❖ Owner can update default payout token

```
function updatePayoutToken(address token) public onlyOwner {  
    dividendTracker.updatePayoutToken(token);  
    emit UpdatePayoutToken(token);  
}
```


- ❖ Owner can change buy and sell fees up to 10%

```
function updateFees(
    uint256 deadBuy,
    uint256 deadSell,
    uint256 marketingBuy,
    uint256 marketingSell,
    uint256 liquidityBuy,
    uint256 liquiditySell,
    uint256 RewardsBuy,
    uint256 RewardsSell,
    uint256 devBuy,
    uint256 devSell
) public onlyOwner {
    buyDeadFees = deadBuy;
    buyMarketingFees = marketingBuy;
    buyLiquidityFee = liquidityBuy;
    buyRewardsFee = RewardsBuy;
    sellDeadFees = deadSell;
    sellMarketingFees = marketingSell;
    sellLiquidityFee = liquiditySell;
    sellRewardsFee = RewardsSell;
    buyDevFee = devBuy;
    sellDevFee = devSell;

    totalSellFees = sellRewardsFee
        .add(sellLiquidityFee)
        .add(sellMarketingFees)
        .add(sellDevFee);

    totalBuyFees = buyRewardsFee
        .add(buyLiquidityFee)
        .add(buyMarketingFees)
        .add(buyDevFee);

    require(totalSellFees <= 10 && totalBuyFees <= 10, "total fees cannot exceed 10% sell or buy");

    emit UpdateFees(
        sellDeadFees,
        sellMarketingFees,
        sellLiquidityFee,
        sellRewardsFee,
        buyDeadFees,
        buyMarketingFees,
        buyLiquidityFee,
        buyRewardsFee,
        buyDevFee,
        sellDevFee
    );
}
```

- ❖ Owner can manually swap and send dividend before swap point reach

```
function forceSwapAndSendDividends(uint256 tokens) public onlyOwner {
    tokens = tokens * (10 ** 18);
    uint256 totalAmount = buyAmount.add(sellAmount);
    uint256 fromBuy = tokens.mul(buyAmount).div(totalAmount);
    uint256 fromSell = tokens.mul(sellAmount).div(totalAmount);

    swapAndSendDividends(tokens);

    buyAmount = buyAmount.sub(fromBuy);
    sellAmount = sellAmount.sub(fromSell);
}
```


- ❖ Owner can airdrop tokens

```
function multiSend(
    address[] memory _contributors,
    uint256[] memory _balances
) public onlyOwner {
    require(
        _contributors.length == _balances.length,
        "Contributors and balances must be same size"
    );
    // Max 200 sends in bulk, uint8 in loop limited to 255
    require(
        _contributors.length <= 200,
        "Contributor list length must be <= 200"
    );
    uint256 sumOfBalances = 0;
    for (uint8 i = 0; i < _balances.length; i++) {
        sumOfBalances = sumOfBalances.add(_balances[i]);
    }
    require(
        balanceOf(msg.sender) >= sumOfBalances,
        "Account balance must be >= sum of balances. "
    );
    require(
        allowance(msg.sender, address(this)) >= sumOfBalances,
        "Contract allowance must be >= sum of balances. "
    );
    address contributor;
    uint256 origBalance;
    for (uint8 j; j < _contributors.length; j++) {
        contributor = _contributors[j];
        require(
            contributor != address(0) &&
            contributor != 0x0000000000000000000000000000000000000000000000000000000000000000,
            "Cannot airdrop to a dead address"
        );
        origBalance = balanceOf(contributor);
        this.transferFrom(msg.sender, contributor, _balances[j]);
        require(
            balanceOf(contributor) == origBalance + _balances[j],
            "Contributor must receive full balance of airdrop"
        );
        emit Airdrop(contributor, _balances[j]);
    }
}
```

Audit conclusion

RugFreeCoins team has performed in-depth testings, 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**

Number of risk issues: **0**

Solidity code functional issue level: **PASS**

Number of owner privileges: **22**

Centralization risk correlated to the active owner: **HIGH**

Smart contract active ownership: **ACTIVE**