RUGFREECOINS

Grok Tether Token

RugfreeCoins Verified on December 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 10%.

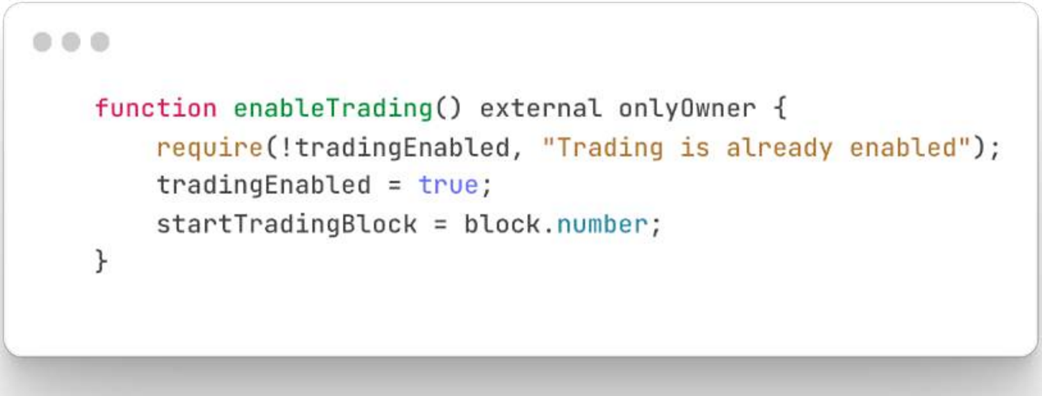
✅ The owner can't blacklist wallets.

✅ 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 is already enabled");
    tradingEnabled = true;
    startTradingBlock = block.number;
}
```

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... 20

Security issue checking status... 21

Owner privileges...23

Audit conclusion... 27

Audit details

Audited project
Grok Tether Token

Contract Address
0x3d2C6366226e044374C5F72Ff9fBd470c395E6E8

Client contact
Grok Tether Token Team

Blockchain
Binance Smart chain

Project website
https://www.groktether.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.

Background

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

https://bscscan.com/token/0x3d2C6366226e044374C5F72Ff9fBd470c395E6E8

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

4% of trade goes to the marketing wallet in BNB
1% of trade goes to the holders in USDT

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

Potential to grow with score points

⚡ Project efficiency	8 / 10
🌟 Project uniqueness	8 / 10
📊 Information quality	8 / 10
👌 Service quality	8 / 10
💻 System quality	8 / 10
🌍 Impact on the community	8 / 10
💼 Impact on the business	9 / 10
🔮 Preparing for the future	8 / 10
🔒 Smart contract security	10/ 10
🛠️ Smart contract functionality assessment	10 / 10
🏆 **Total Score**	**8.5**/ **10**

Contract details

Token contract details for 26th of December 2023

Contract name	**Grok Tether**
Contract address	**0x3d2C6366226e044374C5F72Ff9fBd470c395E6E8**
Token supply	**1,000,000,000**
Token ticker	**$GROKTETHER**
Decimals	**18**
Token holders	**2**
Transaction count	**2**
Contract deployer address	**0xcE399FbD1dda7FdDE7474BBd3D464242bc9a3ABF**
Contract’s current owner address	**0xcE399FbD1dda7FdDE7474BBd3D464242bc9a3ABF**
Marketing wallet	**0xcE399FbD1dda7FdDE7474BBd3D464242bc9a3ABF**
Dividend Tracker	**0x3042C711226EF9e88644415b9F8D2dc2D79193Df**

Contract code function details

№	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	LOW
		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
		Self destruct function security	PASS
3	Business security & centralisation	Access control of owners	HIGH & LOW ISSUES
		Business logics	PASS
		Business implementation	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**		
└	Function Name	Visibility	Mutability	Modifiers
SafeMath	**Library**			
└	tryAdd	Internal 🔒		
└	trySub	Internal 🔒		
└	tryMul	Internal 🔒		
└	tryDiv	Internal 🔒		
└	tryMod	Internal 🔒		
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	sub	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
Context	**Implementation**			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
Ownable	**Implementation**	**Context**		
└		Public ❗	🛑	NO ❗

└	owner	Public ❗		NO ❗
└	_checkOwner	Internal 🔒		
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_transferOwnership	Internal 🔒	🛑	
IERC20	**Interface**			
└	totalSupply	External ❗		NO ❗
└	balanceOf	External ❗		NO ❗
└	transfer	External ❗	🛑	NO ❗
└	allowance	External ❗		NO ❗
└	approve	External ❗	🛑	NO ❗
└	transferFrom	External ❗	🛑	NO ❗
IERC20 Metadata	**Interface**	**IERC20**		
└	name	External ❗		NO ❗
└	symbol	External ❗		NO ❗
└	decimals	External ❗		NO ❗
ERC20	**Implementation**	**Context, IERC20, IERC20 Metadata**		
└		Public ❗	🛑	NO ❗
└	name	Public ❗		NO ❗
└	symbol	Public ❗		NO ❗
└	decimals	Public ❗		NO ❗
└	totalSupply	Public ❗		NO ❗
└	balanceOf	Public ❗		NO ❗
└	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	_spendAllowance	Internal 🔒	🛑	
L	_beforeTokenTransfer	Internal 🔒	🛑	
L	_afterTokenTransfer	Internal 🔒	🛑	
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 🔒		
Iterable Mapping	**Library**			
L	get	Public ❗		NO ❗
L	getIndexOfKey	Public ❗		NO ❗
L	getKeyAtIndex	Public ❗		NO ❗
L	size	Public ❗		NO ❗

└	set	Public ❗	🛑	NO ❗
└	remove	Public ❗	🛑	NO ❗
IUniswapV2 Factory	**Interface**			
└	feeTo	External ❗		NO ❗
└	feeToSetter	External ❗		NO ❗
└	getPair	External ❗		NO ❗
└	allPairs	External ❗		NO ❗
└	allPairsLength	External ❗		NO ❗
└	createPair	External ❗	🛑	NO ❗
└	setFeeTo	External ❗	🛑	NO ❗
└	setFeeToSetter	External ❗	🛑	NO ❗
IUniswapV2 Pair	**Interface**			
└	name	External ❗		NO ❗
└	symbol	External ❗		NO ❗
└	decimals	External ❗		NO ❗
└	totalSupply	External ❗		NO ❗
└	balanceOf	External ❗		NO ❗
└	allowance	External ❗		NO ❗
└	approve	External ❗	🛑	NO ❗
└	transfer	External ❗	🛑	NO ❗
└	transferFrom	External ❗	🛑	NO ❗
└	DOMAIN_SEPARATOR	External ❗		NO ❗
└	PERMIT_TYPEHASH	External ❗		NO ❗
└	nonces	External ❗		NO ❗
└	permit	External ❗	🛑	NO ❗
└	MINIMUM_LIQUIDITY	External ❗		NO ❗

└	factory	External ❗		NO ❗
└	token0	External ❗		NO ❗
└	token1	External ❗		NO ❗
└	getReserves	External ❗		NO ❗
└	price0CumulativeLast	External ❗		NO ❗
└	price1CumulativeLast	External ❗		NO ❗
└	kLast	External ❗		NO ❗
└	mint	External ❗	🛑	NO ❗
└	burn	External ❗	🛑	NO ❗
└	swap	External ❗	🛑	NO ❗
└	skim	External ❗	🛑	NO ❗
└	sync	External ❗	🛑	NO ❗
└	initialize	External ❗	🛑	NO ❗
IUniswapV2 Router01	**Interface**			
└	factory	External ❗		NO ❗
└	WETH	External ❗		NO ❗
└	addLiquidity	External ❗	🛑	NO ❗
└	addLiquidityETH	External ❗	💵	NO ❗
└	removeLiquidity	External ❗	🛑	NO ❗
└	removeLiquidityETH	External ❗	🛑	NO ❗
└	removeLiquidityWithPermit	External ❗	🛑	NO ❗
└	removeLiquidityETHWithPermit	External ❗	🛑	NO ❗
└	swapExactTokensForTokens	External ❗	🛑	NO ❗
└	swapTokensForExactTokens	External ❗	🛑	NO ❗
└	swapExactETHForTokens	External ❗	💵	NO ❗
└	swapTokensForExactETH	External ❗	🛑	NO ❗
└	swapExactTokensForETH	External ❗	🛑	NO ❗
└	swapETHForExactTokens	External ❗	💵	NO ❗

└	quote	External ❗		NO ❗
└	getAmountOut	External ❗		NO ❗
└	getAmountIn	External ❗		NO ❗
└	getAmountsOut	External ❗		NO ❗
└	getAmountsIn	External ❗		NO ❗
IUniswapV2 Router02	**Interface**	**IUniswapV2 Router01**		
└	removeLiquidityETHSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO ❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
Dividend PayingToken Interface	**Interface**			
└	dividendOf	External ❗		NO ❗
└	withdrawDividend	External ❗	🛑	NO ❗
Dividend PayingToken Optional Interface	**Interface**			
└	withdrawableDividendOf	External ❗		NO ❗
└	withdrawnDividendOf	External ❗		NO ❗
└	accumulativeDividendOf	External ❗		NO ❗
Dividend PayingToken	**Implementation**	**ERC20, Ownable, Dividend**		

		PayingToken Interface, Dividend PayingToken Optional Interface		
└		Public ❗️	🛑	ERC20
└	distributeDividends	Public ❗️	🛑	onlyOwner
└	withdrawDividend	Public ❗️	🛑	NO ❗️
└	_withdrawDividendOfUser	Internal 🔒	🛑	
└	dividendOf	Public ❗️		NO ❗️
└	withdrawableDividendOf	Public ❗️		NO ❗️
└	withdrawnDividendOf	Public ❗️		NO ❗️
└	accumulativeDividendOf	Public ❗️		NO ❗️
└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_setBalance	Internal 🔒	🛑	
Dividend Tracker	**Implementation**	**Ownable, Dividend PayingToken**		
└		Public ❗️	🛑	Dividend Paying Token
└	_transfer	Internal 🔒		
└	withdrawDividend	Public ❗️		NO ❗️
└	updateMinimumTokenBalanceForDividends	External ❗️	🛑	onlyOwner
└	excludeFromDividends	External ❗️	🛑	onlyOwner
└	updateClaimWait	External ❗️	🛑	onlyOwner
└	setLastProcessedIndex	External ❗️	🛑	onlyOwner
└	getLastProcessedIndex	External ❗️		NO ❗️
└	getNumberOfTokenHolders	External ❗️		NO ❗️

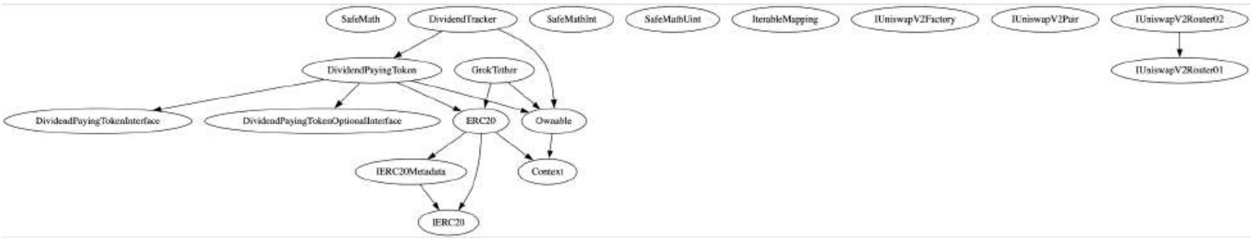
└	getAccount	Public ❗		NO ❗
└	getAccountAtIndex	Public ❗		NO ❗
└	canAutoClaim	Private 🔐		
└	setBalance	External ❗	🛑	onlyOwner
└	process	Public ❗	🛑	NO ❗
└	processAccount	Public ❗	🛑	onlyOwner
GrokTether	**Implementation**	**ERC20, Ownable**		
└		Public ❗	💵	ERC20
└		External ❗	💵	NO ❗
└	withdrawStuckTokens	External ❗	🛑	onlyOwner
└	withdrawStuckETH	External ❗	🛑	onlyOwner
└	isContract	Internal 🔒		
└	_setAutomatedMarketMakerPair	Private 🔐	🛑	
└	updateBuyFees	External ❗	🛑	onlyOwner
└	updateSellFees	External ❗	🛑	onlyOwner
└	setWhitelistStatus	External ❗	🛑	onlyOwner
└	iswhitelisted	Public ❗		NO ❗
└	setmarketingWallet	External ❗	🛑	onlyOwner
└	enableTrading	External ❗	🛑	onlyOwner
└	_transfer	Internal 🔒	🛑	
└	swapAndSendDividends	Private 🔐	🛑	
└	setSwapTokensAtAmount	External ❗	🛑	onlyOwner
└	updateGasForProcessing	Public ❗	🛑	onlyOwner
└	updateMinimumBalanceForDividends	External ❗	🛑	onlyOwner
└	updateClaimWait	External ❗	🛑	onlyOwner
└	getClaimWait	External ❗		NO ❗
└	getTotalDividendsDistributed	External ❗		NO ❗
└	withdrawableDividendOf	Public ❗		NO ❗

L	dividendTokenBalanceOf	Public ❗		NO ❗
L	totalRewardsEarned	Public ❗		NO ❗
L	excludeFromDividends	External ❗	🛑	onlyOwner
L	getAccountDividendsInfo	External ❗		NO ❗
L	getAccountDividendsInfoAtIndex	External ❗		NO ❗
L	processDividendTracker	External ❗	🛑	NO ❗
L	claim	External ❗	🛑	NO ❗
L	claimAddress	External ❗	🛑	onlyOwner
L	getLastProcessedIndex	External ❗		NO ❗
L	setLastProcessedIndex	External ❗	🛑	onlyOwner
L	getNumberOfDividendTokenHolders	External ❗		NO ❗

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

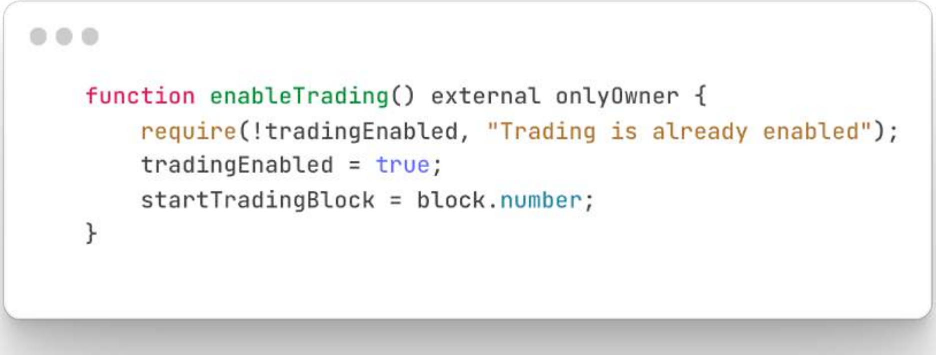
Inheritance Hierarchy



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.



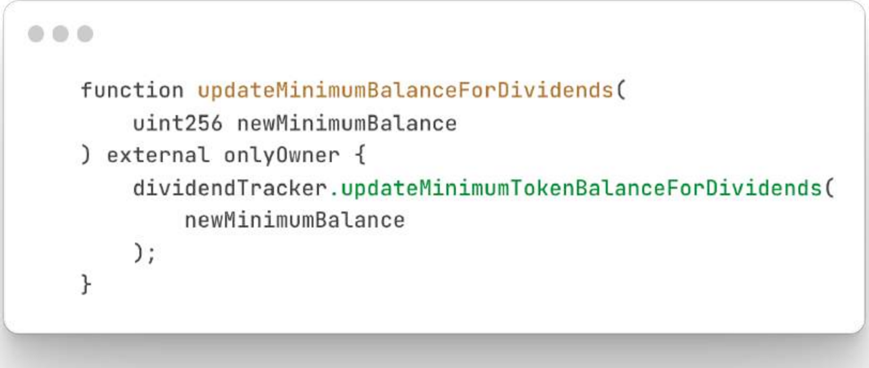
```
function enableTrading() external onlyOwner {
    require(!tradingEnabled, "Trading is already enabled");
    tradingEnabled = true;
    startTradingBlock = block.number;
}
```

❖ Medium severity issues

No medium severity issues found

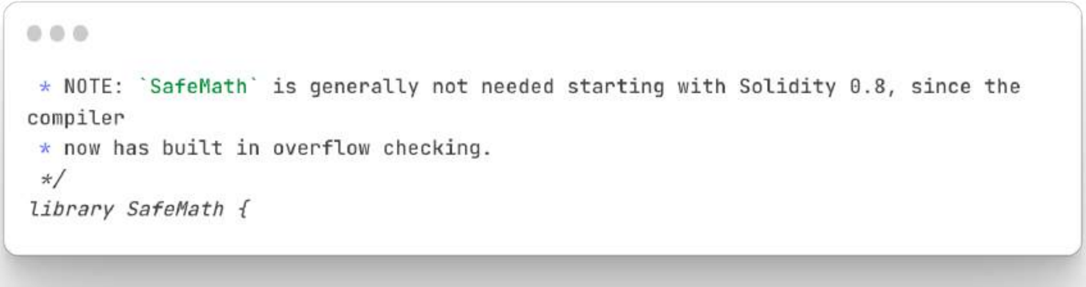
❖ Low severity issues

The owner can alter the minimum token threshold required to receive rewards without any limitations. If the owner sets this threshold to the total supply, no one will be eligible to receive rewards.



```
function updateMinimumBalanceForDividends(
    uint256 newMinimumBalance
) external onlyOwner {
    dividendTracker.updateMinimumTokenBalanceForDividends(
        newMinimumBalance
    );
}
```

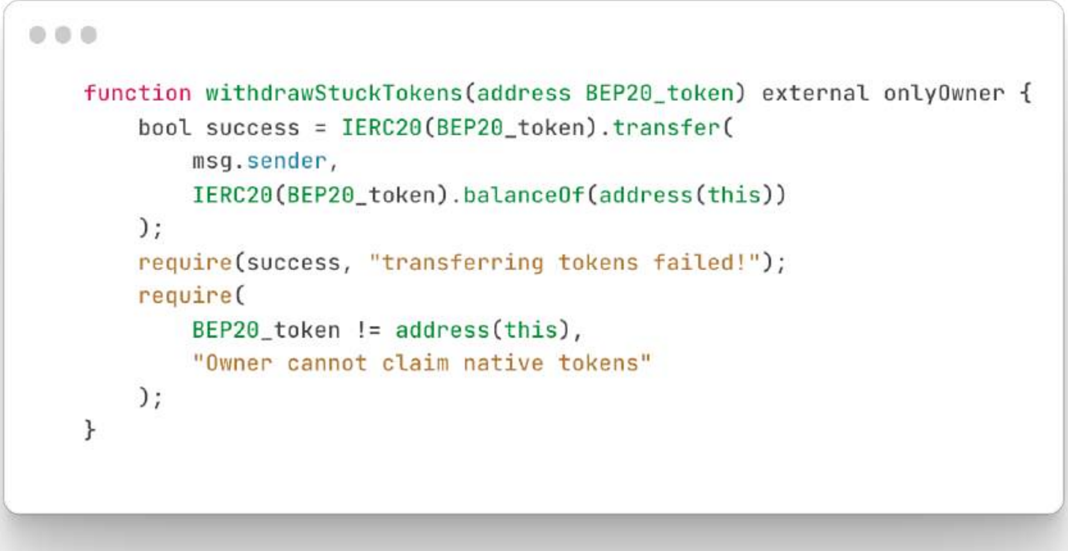
The contract is using the SafeMath library, but starting from Solidity 0.8.0 onward, SafeMath is built-in.



```
 * NOTE: `SafeMath` is generally not needed starting with Solidity 0.8, since the
compiler
 * now has built in overflow checking.
 */
library SafeMath {
```

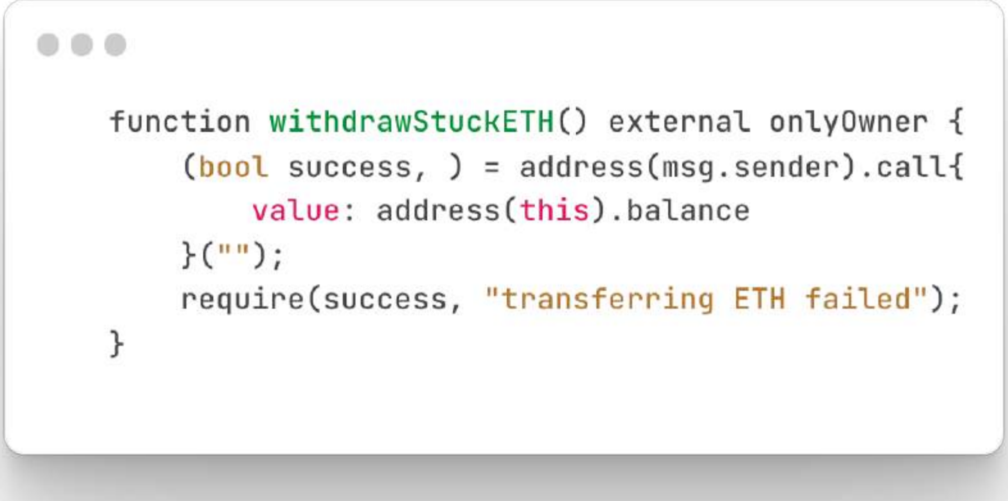
Owner privileges

- Owner can get any bep20 tokens from the contract (can not get native tokens)



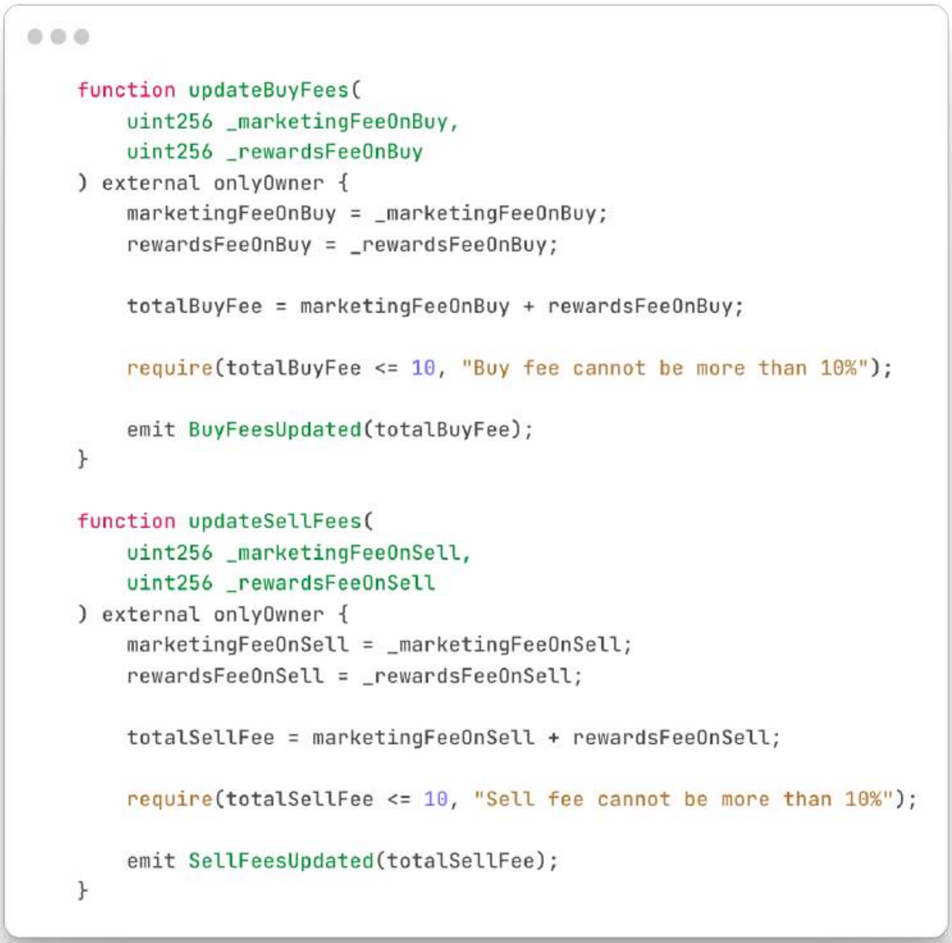
```solidity
function withdrawStuckTokens(address BEP20_token) external onlyOwner {
    bool success = IERC20(BEP20_token).transfer(
        msg.sender,
        IERC20(BEP20_token).balanceOf(address(this))
    );
    require(success, "transferring tokens failed!");
    require(
        BEP20_token != address(this),
        "Owner cannot claim native tokens"
    );
}
```

- Owner can withdraw BNB from the contract



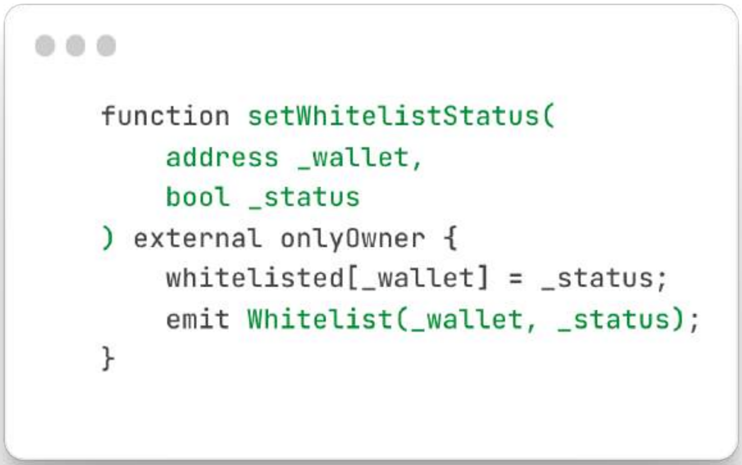
```solidity
function withdrawStuckETH() external onlyOwner {
    (bool success, ) = address(msg.sender).call{
        value: address(this).balance
    }("");
    require(success, "transferring ETH failed");
}
```

- Owner can change all buy and sell fees maximum up to 20% (each 10% max)



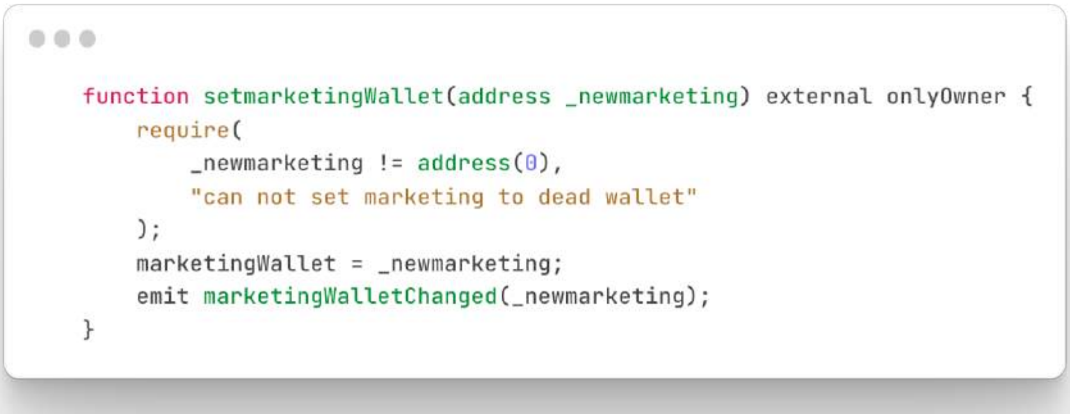
```
function updateBuyFees(
    uint256 _marketingFeeOnBuy,
    uint256 _rewardsFeeOnBuy
) external onlyOwner {
    marketingFeeOnBuy = _marketingFeeOnBuy;
    rewardsFeeOnBuy = _rewardsFeeOnBuy;

    totalBuyFee = marketingFeeOnBuy + rewardsFeeOnBuy;

    require(totalBuyFee <= 10, "Buy fee cannot be more than 10%");

    emit BuyFeesUpdated(totalBuyFee);
}

function updateSellFees(
    uint256 _marketingFeeOnSell,
    uint256 _rewardsFeeOnSell
) external onlyOwner {
    marketingFeeOnSell = _marketingFeeOnSell;
    rewardsFeeOnSell = _rewardsFeeOnSell;

    totalSellFee = marketingFeeOnSell + rewardsFeeOnSell;

    require(totalSellFee <= 10, "Sell fee cannot be more than 10%");

    emit SellFeesUpdated(totalSellFee);
}
```

- Owner can include/exclude wallets from fees



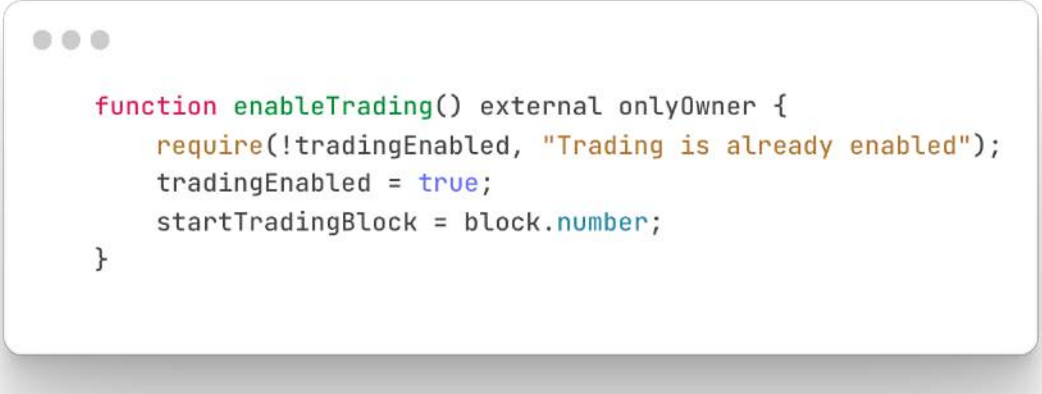
```
function setWhitelistStatus(
    address _wallet,
    bool _status
) external onlyOwner {
    whitelisted[_wallet] = _status;
    emit Whitelist(_wallet, _status);
}
```

- Owner can change marketing wallet



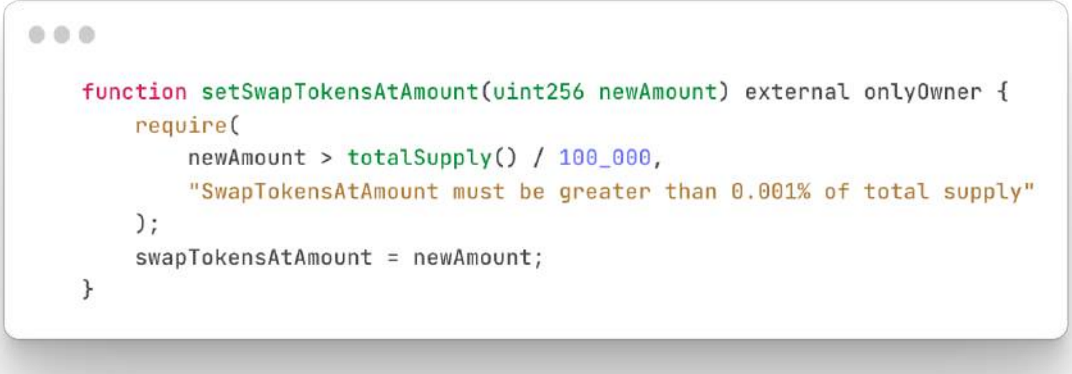
```
function setmarketingWallet(address _newmarketing) external onlyOwner {
    require(
        _newmarketing != address(0),
        "can not set marketing to dead wallet"
    );
    marketingWallet = _newmarketing;
    emit marketingWalletChanged(_newmarketing);
}
```

- Owner can enable trading,once enabled can not disable again



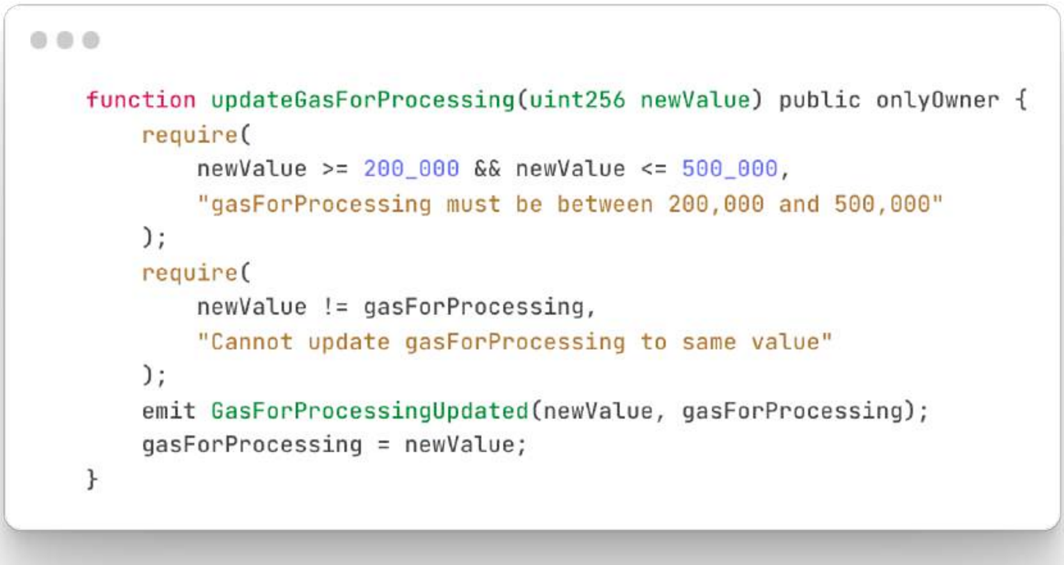
```
function enableTrading() external onlyOwner {
    require(!tradingEnabled, "Trading is already enabled");
    tradingEnabled = true;
    startTradingBlock = block.number;
}
```

- Owner can change swap point minimum up to 0.001%



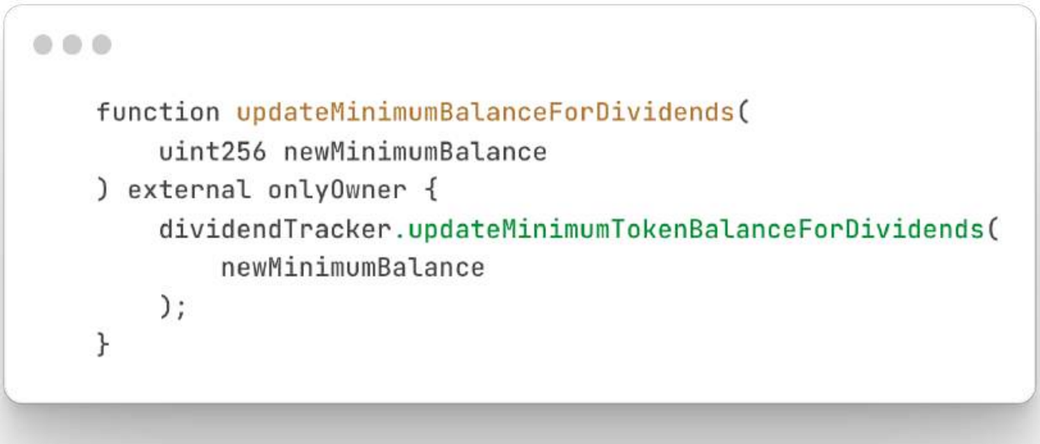
```
function setSwapTokensAtAmount(uint256 newAmount) external onlyOwner {
    require(
        newAmount > totalSupply() / 100_000,
        "SwapTokensAtAmount must be greater than 0.001% of total supply"
    );
    swapTokensAtAmount = newAmount;
}
```

- Owner can change maximum gas usage to process reward tracker at a time



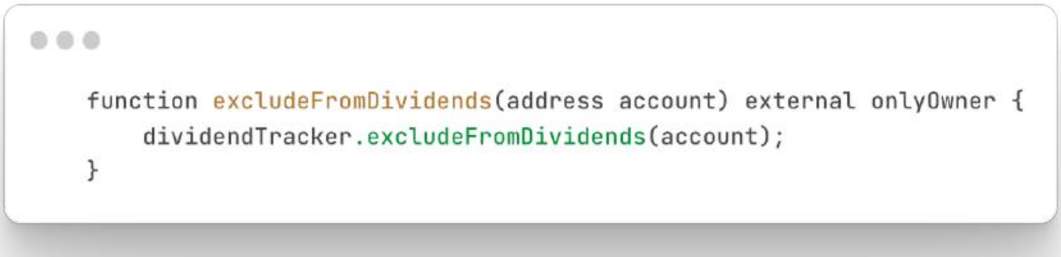
```
function updateGasForProcessing(uint256 newValue) public onlyOwner {
    require(
        newValue >= 200_000 && newValue <= 500_000,
        "gasForProcessing must be between 200,000 and 500,000"
    );
    require(
        newValue != gasForProcessing,
        "Cannot update gasForProcessing to same value"
    );
    emit GasForProcessingUpdated(newValue, gasForProcessing);
    gasForProcessing = newValue;
}
```

- Owner can change minimum token amount to hold to receive rewards



```
function updateMinimumBalanceForDividends(
    uint256 newMinimumBalance
) external onlyOwner {
    dividendTracker.updateMinimumTokenBalanceForDividends(
        newMinimumBalance
    );
}
```

- Owner can include/exclude wallets from fees



```
function excludeFromDividends(address account) external onlyOwner {
    dividendTracker.excludeFromDividends(account);
}
```

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 & LOW ISSUES ▾
Number of risk issues:	3
Solidity code functional issue level:	PASS ▾
Number of owner privileges:	10
Centralization risk correlated to the active owner:	HIGH ▾
Smart contract active ownership:	ACTIVE ▾