RugFreeCoins Audit

Probot Token
Smart Contract Security Audit

July 30th ,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.

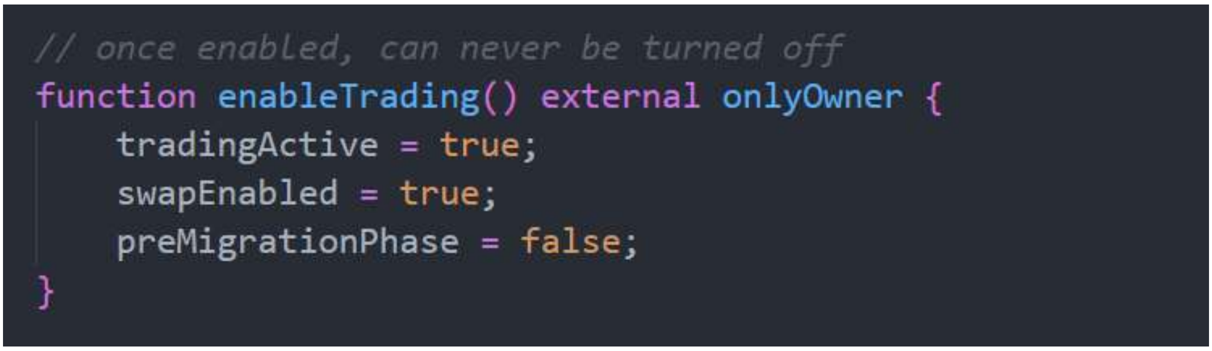
✅ The owner can't blacklist wallets.

✅ The owner can’t set a max wallet limit

❌ The owner can’t claim the contract’s balance of its own token. (The owner can can claim Probot tokens from the contract)

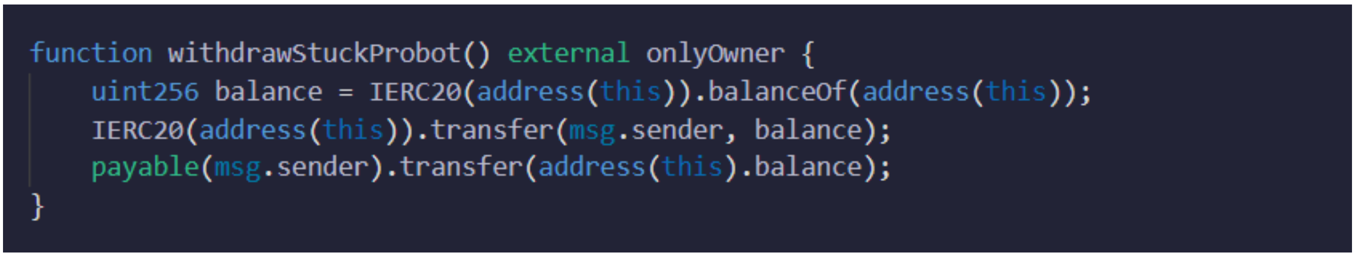
- **HIGH SEVERITY ISSUES**

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



```
// once enabled, can never be turned off
function enableTrading() external onlyOwner {
    tradingActive = true;
    swapEnabled = true;
    preMigrationPhase = false;
}
```

Owner can withdraw native tokens from the contract



```
function withdrawStuckProbot() external onlyOwner {
    uint256 balance = IERC20(address(this)).balanceOf(address(this));
    IERC20(address(this)).transfer(msg.sender, balance);
    payable(msg.sender).transfer(address(this).balance);
}
```

Contents

Audit details

Audited project
Probot Token

Contract Address
0x07781FbBc0EB7fDA846b923d3C45eDCb0F70d466

Client contact
Probot Token Team

Blockchain
Ethereum

Project website
https://probot-hub.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 Probot Token Team to perform an audit of the smart contract.

https://etherscan.io/address/0x07781fbbc0eb7fda846b923d3c45edcb0f70d466

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% of trade goes to the team wallet in ETH
- 1% of trade goes to the Rev share wallet in ETH
- 1% of trade goes to the liquidity pool

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

Potential to grow with score points

1.	Project efficiency	8/10
2.	Project uniqueness	7/10
3	Information quality	8/10
4	Service quality	8/10
5	System quality	8/10
6	Impact on the community	8/10
7	Impact on the business	8/10
8	Preparing for the future	8/10
9	Smart contract security	8/10
10	Smart contract functionality assessment	10/10
Total Points		**8.1/10**

Contract details

Token contract details for 30th of July 2023

Contract name	Probot
Contract address	0x07781FbBc0EB7fDA846b923d3C45eDCb0F70d466
Token supply	1,000,000
Token ticker	PROBOT
Decimals	18
Token holders	1
Transaction count	1
Contract deployer address	0x8C9bAFB13A333d3430809258639FE7a49fab067b
Contract ‘s current owner address	0x8C9bAFB13A333d3430809258639FE7a49fab067b
Rev share wallet	0x6Cd67D034d39a82d9C12B6E14FDffDfD9ce68997
Team wallet	0x8C9bAFB13A333d3430809258639FE7a49fab067b

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	HIGH ISSUE
		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 🔒		
Ownable	**Implementation**	**Context**		
L		Public ❗	🛑	NO ❗
L	owner	Public ❗		NO ❗
L	renounceOwnership	Public ❗	🛑	onlyOwner
L	transferOwnership	Public ❗	🛑	onlyOwner
L	_transferOwnership	Internal 🔒	🛑	
IERC20	**Interface**			
L	totalSupply	External ❗		NO ❗
L	balanceOf	External ❗		NO ❗
L	transfer	External ❗	🛑	NO ❗
L	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❗
└	allowance	Public ❗		NO❗
└	approve	Public ❗	🛑	NO❗
└	transferFrom	Public ❗	🛑	NO❗
└	increaseAllowance	Public ❗	🛑	NO❗
└	decreaseAllowance	Public ❗	🛑	NO❗

L	_transfer	Internal 🔒	🛑	
L	_mint	Internal 🔒	🛑	
L	_burn	Internal 🔒	🛑	
L	_approve	Internal 🔒	🛑	
L	_beforeTokenTransfer	Internal 🔒	🛑	
L	_afterTokenTransfer	Internal 🔒	🛑	
SafeMath	**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 🔒		

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❗
IUniswapV2 Pair	**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 Router02	**Interface**			
L	factory	External ❗		NO ❗

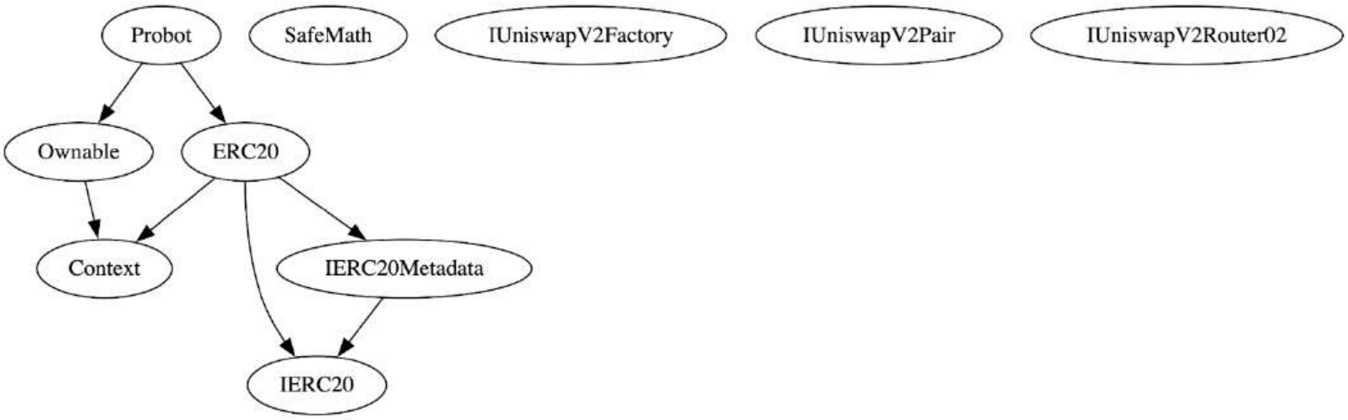
L	WETH	External ❗		NO ❗
L	addLiquidity	External ❗	🛑	NO ❗
L	addLiquidityETH	External ❗	💵	NO ❗
L	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
L	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO ❗
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
Probot	**Implementation**	**ERC20, Ownable**		
L		Public ❗	🛑	ERC20
L		External ❗	💵	NO ❗
L	enableTrading	External ❗	🛑	onlyOwner
L	updateSwapTokensAtAmount	External ❗	🛑	onlyOwner
L	updateSwapEnabled	External ❗	🛑	onlyOwner
L	updateBuyFees	External ❗	🛑	onlyOwner
L	updateSellFees	External ❗	🛑	onlyOwner
L	excludeFromFees	Public ❗	🛑	onlyOwner
L	setAutomatedMarketMakerPair	Public ❗	🛑	onlyOwner
L	_setAutomatedMarketMakerPair	Private 🔐	🛑	
L	updateRevShareWallet	External ❗	🛑	onlyOwner
L	updateTeamWallet	External ❗	🛑	onlyOwner
L	isExcludedFromFees	Public ❗		NO ❗

L	_transfer	Internal 🔒	🛑	
L	swapTokensForEth	Private 🔐	🛑	
L	addLiquidity	Private 🔐	🛑	
L	swapBack	Private 🔐	🛑	
L	withdrawStuckProbot	External ❗	🛑	onlyOwner
L	withdrawStuckEth	External ❗	🛑	onlyOwner
L	setPreMigrationTransferable	Public ❗	🛑	onlyOwner

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

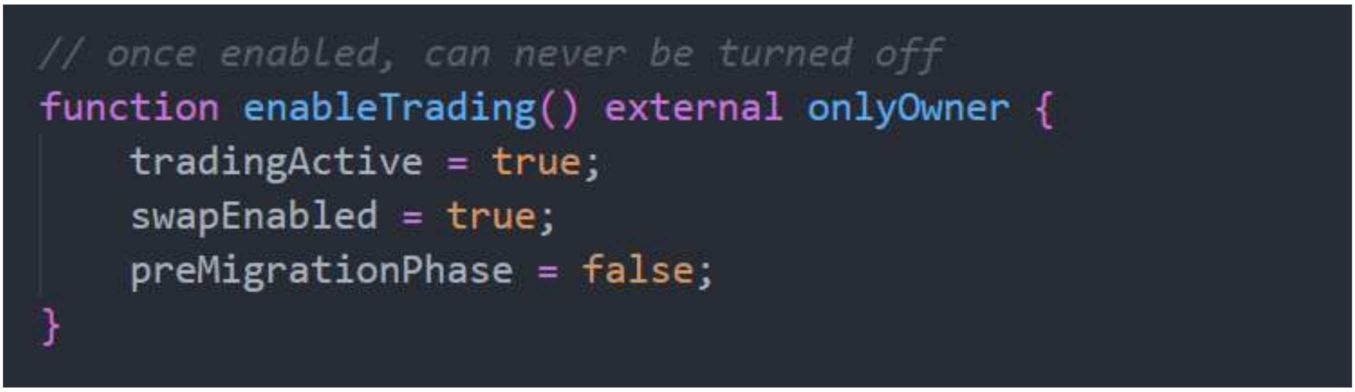
Inheritance Hierarchy



Security issue checking status

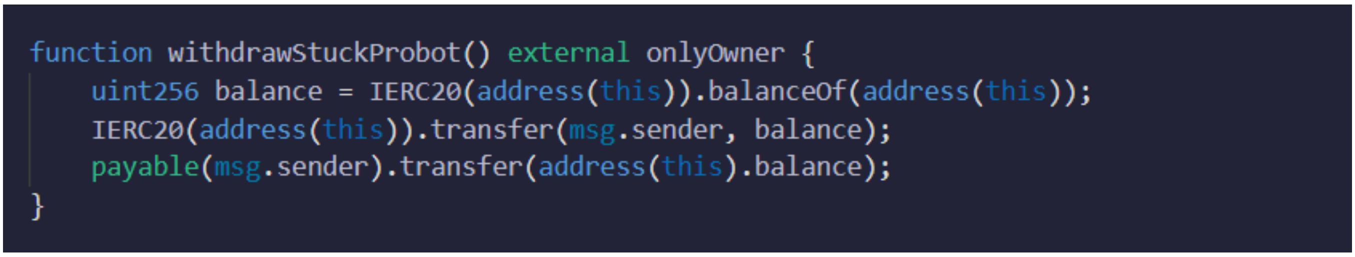
- ❖ **HIGH SEVERITY ISSUES**

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



```
// once enabled, can never be turned off
function enableTrading() external onlyOwner {
    tradingActive = true;
    swapEnabled = true;
    preMigrationPhase = false;
}
```

Owner can withdraw native tokens from the contract



```
function withdrawStuckProbot() external onlyOwner {
    uint256 balance = IERC20(address(this)).balanceOf(address(this));
    IERC20(address(this)).transfer(msg.sender, balance);
    payable(msg.sender).transfer(address(this).balance);
}
```

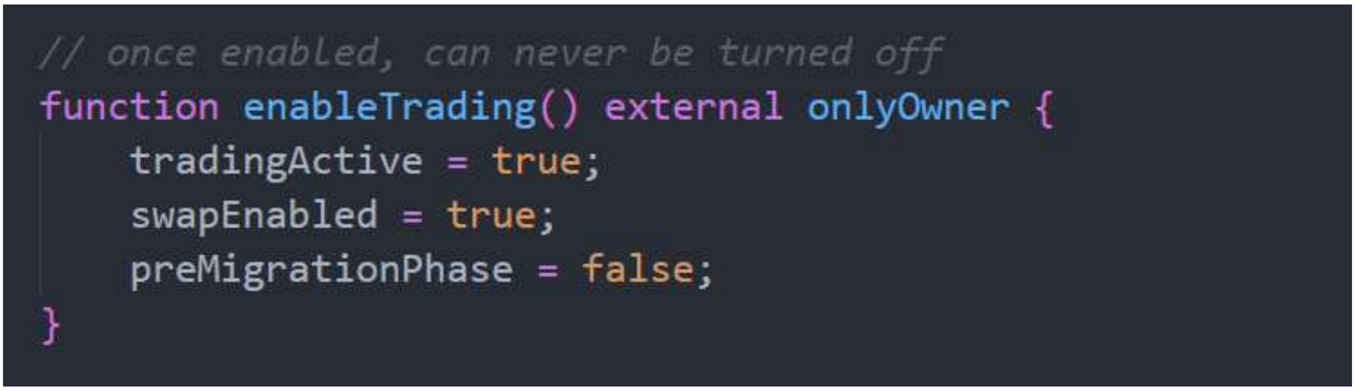
- ❖ **MEDIUM SEVERITY ISSUES**
  No medium severity issues found

- ❖ **LOW SEVERITY ISSUES**
  No low-severity issues found

- ❖ **MEDIUM SEVERITY ISSUES**
  No centralization issues found

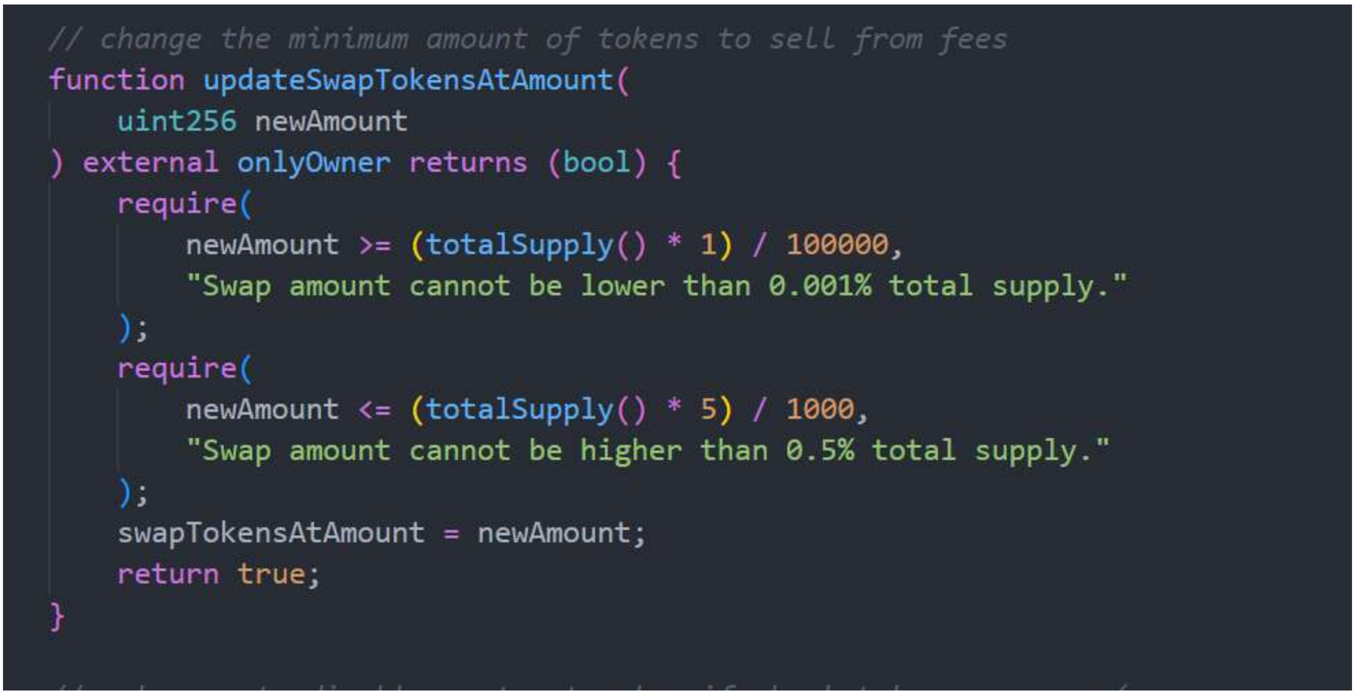
Owner privileges

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



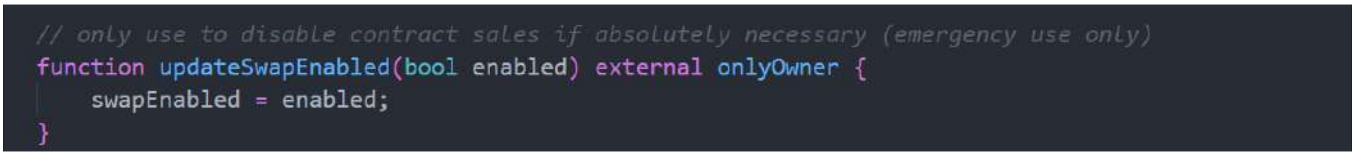
```
// once enabled, can never be turned off
function enableTrading() external onlyOwner {
    tradingActive = true;
    swapEnabled = true;
    preMigrationPhase = false;
}
```

- Owner can change swap point between 0.001%-0.5%



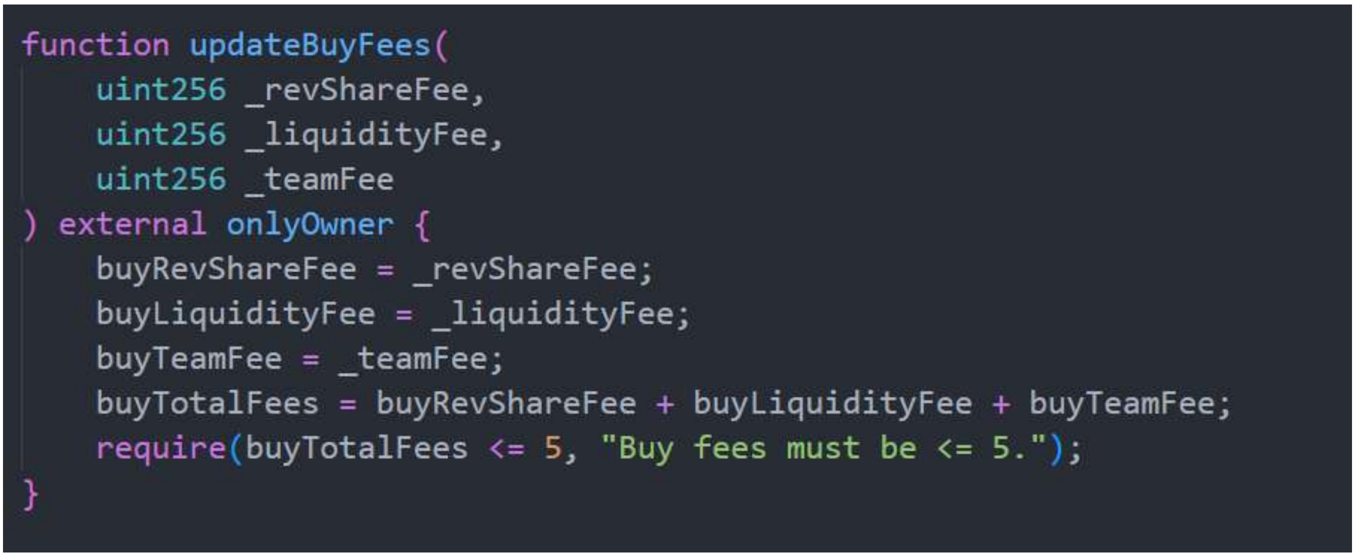
```
// change the minimum amount of tokens to sell from fees
function updateSwapTokensAtAmount(
    uint256 newAmount
) external onlyOwner returns (bool) {
    require(
        newAmount >= (totalSupply() * 1) / 100000,
        "Swap amount cannot be lower than 0.001% total supply."
    );
    require(
        newAmount <= (totalSupply() * 5) / 1000,
        "Swap amount cannot be higher than 0.5% total supply."
    );
    swapTokensAtAmount = newAmount;
    return true;
}
```

- Owner can enable/disable swapping



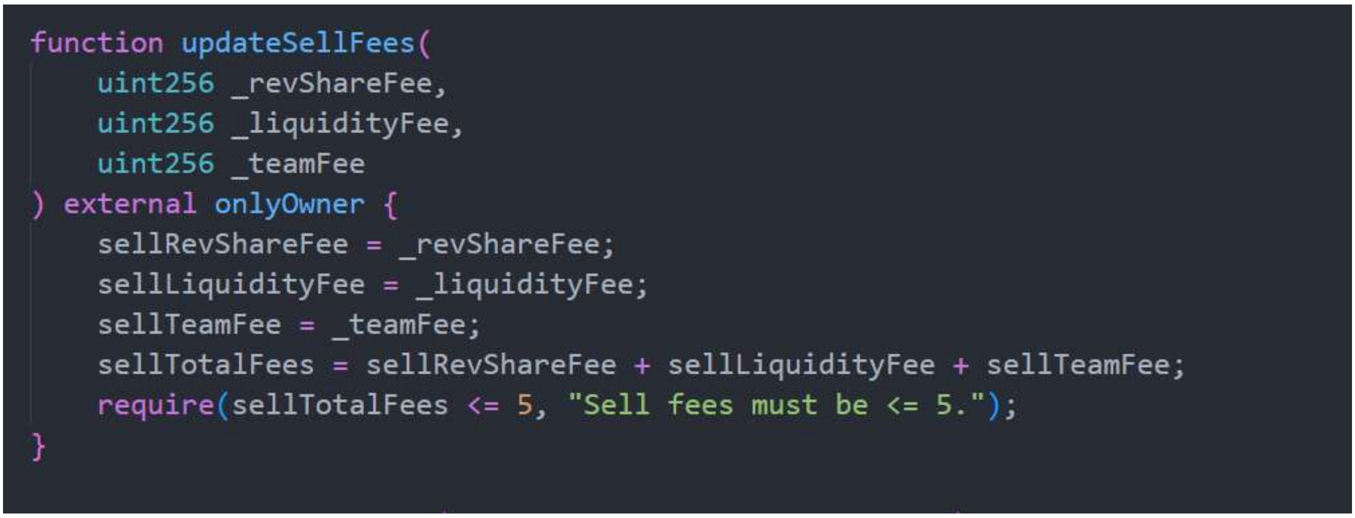
```
// only use to disable contract sales if absolutely necessary (emergency use only)
function updateSwapEnabled(bool enabled) external onlyOwner {
    swapEnabled = enabled;
}
```

- Owner can change all buy fees maximum up to 5%



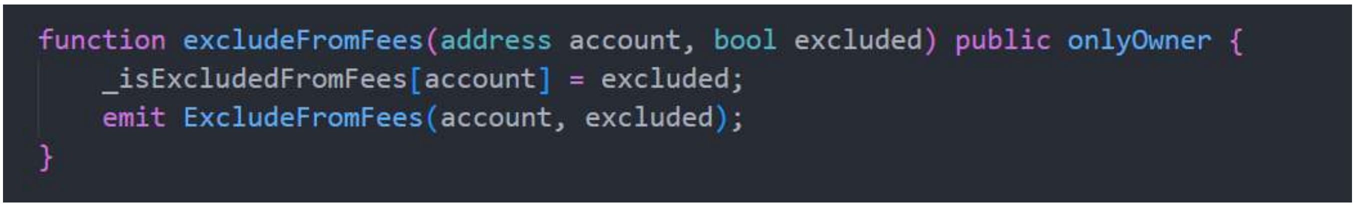
```
function updateBuyFees(
    uint256 _revShareFee,
    uint256 _liquidityFee,
    uint256 _teamFee
) external onlyOwner {
    buyRevShareFee = _revShareFee;
    buyLiquidityFee = _liquidityFee;
    buyTeamFee = _teamFee;
    buyTotalFees = buyRevShareFee + buyLiquidityFee + buyTeamFee;
    require(buyTotalFees <= 5, "Buy fees must be <= 5.");
}
```

- Owner can change all sell fees maximum upto 5%



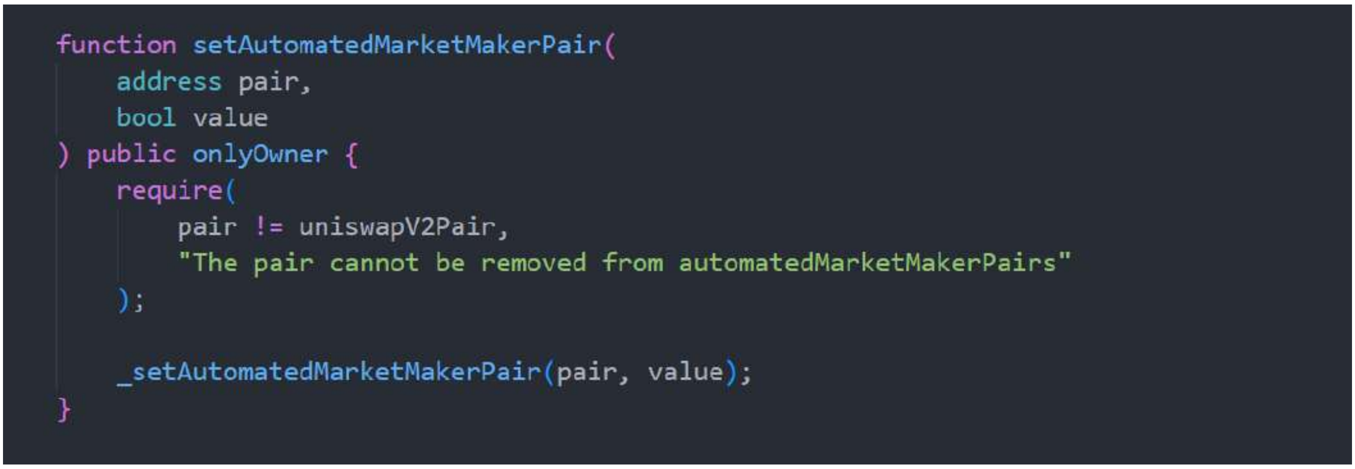
```
function updateSellFees(
    uint256 _revShareFee,
    uint256 _liquidityFee,
    uint256 _teamFee
) external onlyOwner {
    sellRevShareFee = _revShareFee;
    sellLiquidityFee = _liquidityFee;
    sellTeamFee = _teamFee;
    sellTotalFees = sellRevShareFee + sellLiquidityFee + sellTeamFee;
    require(sellTotalFees <= 5, "Sell fees must be <= 5.");
}
```

- Owner can include/exclude wallets from fees



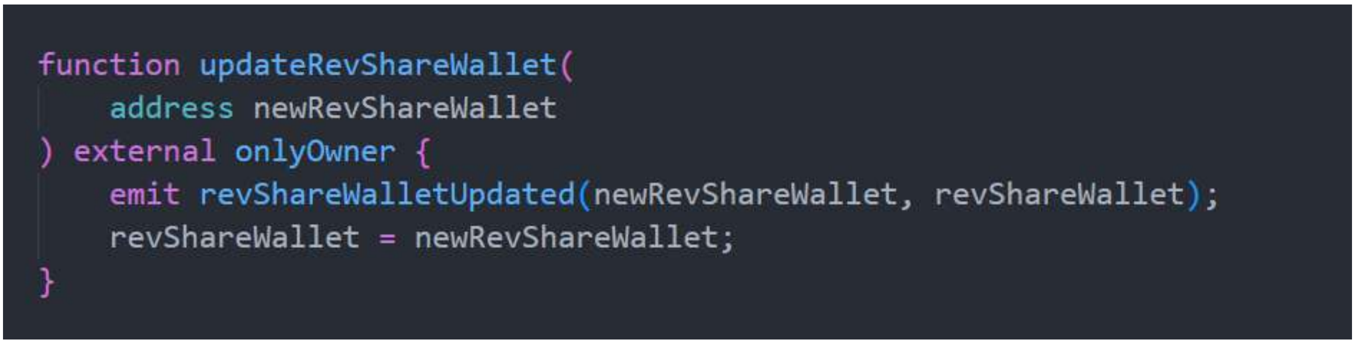
```
function excludeFromFees(address account, bool excluded) public onlyOwner {
    _isExcludedFromFees[account] = excluded;
    emit ExcludeFromFees(account, excluded);
}
```

- Owner can add/remove LP Paires



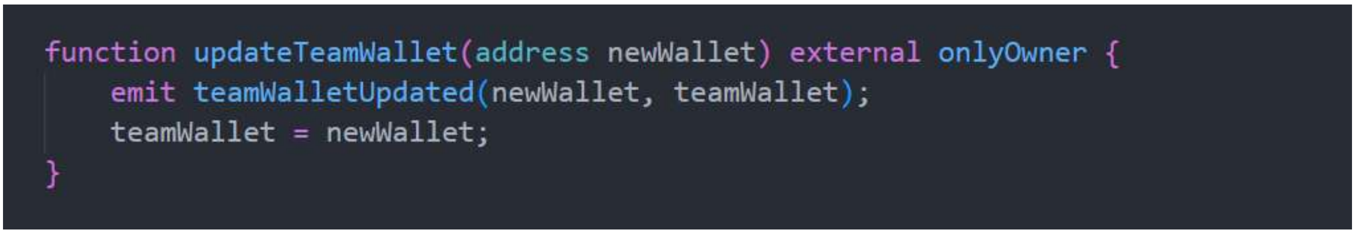
```
function setAutomatedMarketMakerPair(
    address pair,
    bool value
) public onlyOwner {
    require(
        pair != uniswapV2Pair,
        "The pair cannot be removed from automatedMarketMakerPairs"
    );

    _setAutomatedMarketMakerPair(pair, value);
}
```

- Owner can change rev share wallet address



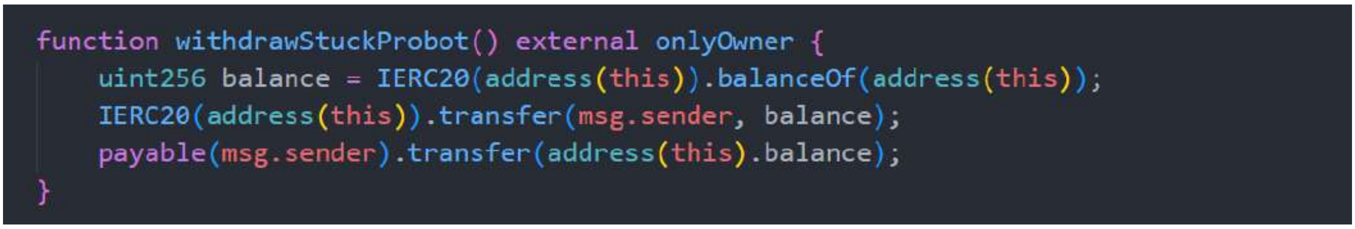
```
function updateRevShareWallet(
    address newRevShareWallet
) external onlyOwner {
    emit revShareWalletUpdated(newRevShareWallet, revShareWallet);
    revShareWallet = newRevShareWallet;
}
```

- Owner can change team wallet address



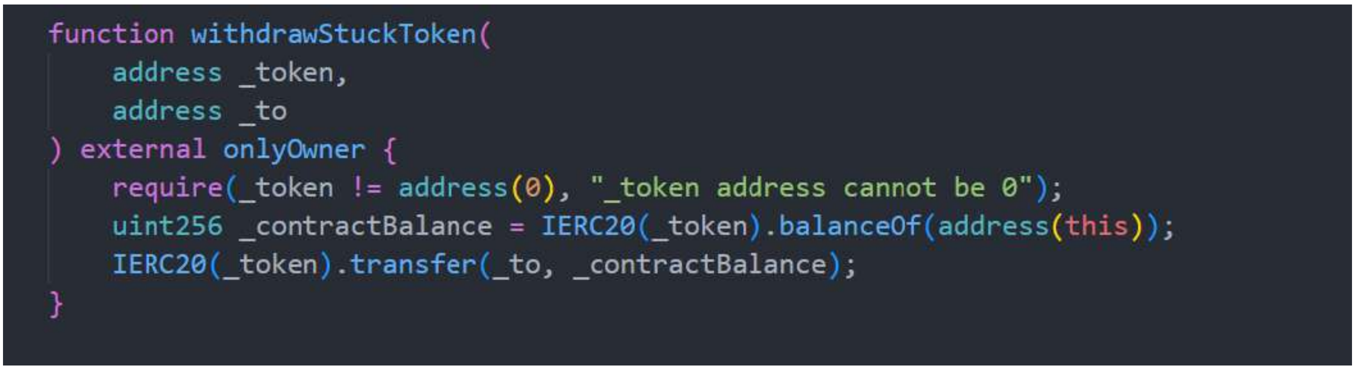
```
function updateTeamWallet(address newWallet) external onlyOwner {
    emit teamWalletUpdated(newWallet, teamWallet);
    teamWallet = newWallet;
}
```

- Owner can withdraw native tokens and ETHfrom the contract



```
function withdrawStuckProbot() external onlyOwner {
    uint256 balance = IERC20(address(this)).balanceOf(address(this));
    IERC20(address(this)).transfer(msg.sender, balance);
    payable(msg.sender).transfer(address(this).balance);
}
```

- Owner can withdraw any erc20 tokens from the contract



```
function withdrawStuckToken(
    address _token,
    address _to
) external onlyOwner {
    require(_token != address(0), "_token address cannot be 0");
    uint256 _contractBalance = IERC20(_token).balanceOf(address(this));
    IERC20(_token).transfer(_to, _contractBalance);
}
```

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: **3**

Solidity code functional issue level: **PASS**

Number of owner privileges: **11**

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

Smart contract active ownership: **ACTIVE**