RugFreeCoins Audit

Bugs Bunny Token Smart Contract Security Audit

July 16th ,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 enable or pause trading.

✅ The owner can't change fees by more than 20%.

✅ 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.

Contents

Audit details

Audited project
Bugs Bunny Token

Contract Address
0x428ACfC53146024a6aB041e9E20Bf5B42537CB62

Client contact
Bugs Bunny Token Team

Blockchain
Binance Smart chain

Project website
https://bbunny.io/

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 Bugs Bunny Token Team to perform an audit of the smart contract.

https://bscscan.com/token/0x428acfc53146024a6ab041e9e20bf5b42537cb62

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.

Roadmap

Quarter 01

- Launch Telegram
- Launch Website
- Verify Contract
- Launch Fairlaunch Presale
- Dextools Socials
- Launch to PCS
- Heavy Marketing
- Ave Trending
- CG and CMC Fast-Tracked

Quarter 02

- Community DAO
- 20,000 Dummys
- Mass adoption
- strategic mainstream marketing
- Unique utility
- Dummys building
- Charity Donations

Tokenomics

4% tax when buying & selling

- 4% of trade goes to the marketing wallet in BNB

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

Potential to grow with score points

1.	Project efficiency	8/10
2.	Project uniqueness	8/10
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	9/10
8	Preparing for the future	8/10
9	Smart contract security	10/10
10	Smart contract functionality assessment	9/10
Total Points		**8.4/10**

Contract details

Token contract details for 16th of July 2023

Contract name	Bugs Bunny
Contract address	0x428ACfC53146024a6aB041e9E20Bf5B42537CB62
Token supply	100,000,000,000
Token ticker	BBY
Decimals	18
Token holders	1
Transaction count	1
Contract deployer address	0x0A5b0CAd8A2ABAa75F3e576C028BCDbF80c279db
Contract's current owner address	0x0a5b0cad8a2abaa75f3e576c028bcdbf80c279db
SAFU Wallet	0x0a5b0cad8a2abaa75f3e576c028bcdbf80c279db
Marketing wallet	0x0a5b0cad8a2abaa75f3e576c028bcdbf80c279db

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	Low issue
		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 🔒		
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❗
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	verifyCallResultFromTarget	Internal 🔒		
L	verifyCallResult	Internal 🔒		
L	_revert	Private 🔐		
Ownable	**Implementation**	**Context**		
L		Public ❗	🛑	NO ❗
L	owner	Public ❗		NO ❗
L	_checkOwner	Internal 🔒		
L	renounceOwnership	Public ❗	🛑	onlyOwner
L	transferOwnership	Public ❗	🛑	onlyOwner
L	_transferOwnership	Internal 🔒	🛑	
IUniswapV2 Factory	**Interface**			
L	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❗
└	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**	**Iuniswap V2 Router01**		
└	removeLiquidityETHSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO ❗

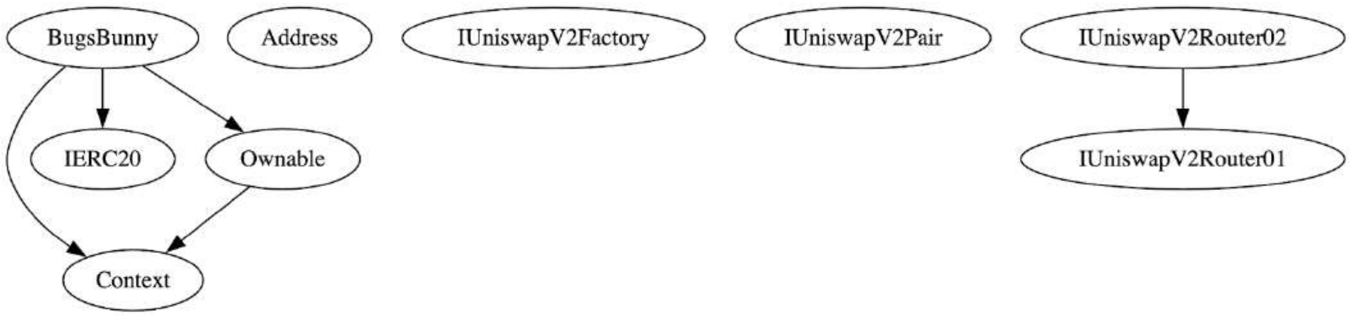
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
BugsBunny	**Implementation**	**Context, IERC20, Ownable**		
└		Public ❗	🛑	NO ❗
└	totalSupply	Public ❗		NO ❗
└	balanceOf	Public ❗		NO ❗
└	transfer	Public ❗	🛑	NO ❗
└	allowance	Public ❗		NO ❗
└	approve	Public ❗	🛑	NO ❗
└	transferFrom	Public ❗	🛑	NO ❗
└	increaseAllowance	Public ❗	🛑	NO ❗
└	decreaseAllowance	Public ❗	🛑	NO ❗
└	_approve	Private 🔐	🛑	
└	_transfer	Private 🔐	🛑	
└	swapAndLiquify	Public ❗	🛑	lockTheSwap
└	swapTokensForEth	Private 🔐	🛑	
└	_tokenTransfer	Private 🔐	🛑	
└	isExcludedFromFee	External ❗		NO ❗
└	excludeFromFee	External ❗	🛑	onlyOwner
└	includeInFee	External ❗	🛑	onlyOwner
└	setTokensToSwap	External ❗	🛑	onlyOwner

L	setSwapAndLiquifyEnabled	External ❗	🛑	onlyOwner
L	setMarketingWallet	External ❗	🛑	onlyOwner
L	setBuyFee	External ❗	🛑	onlyOwner
L	setSellFee	External ❗	🛑	onlyOwner
L	transferToAddressETH	Private 🔐	🛑	
L		External ❗	💵	NO ❗
L	swapETHForTokens	Private 🔐	🛑	
L	recoverETHfromContract	External ❗	🛑	onlyOwner
L	recoverTokensFromContract	External ❗	🛑	onlyOwner

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

Inheritance Hierarchy



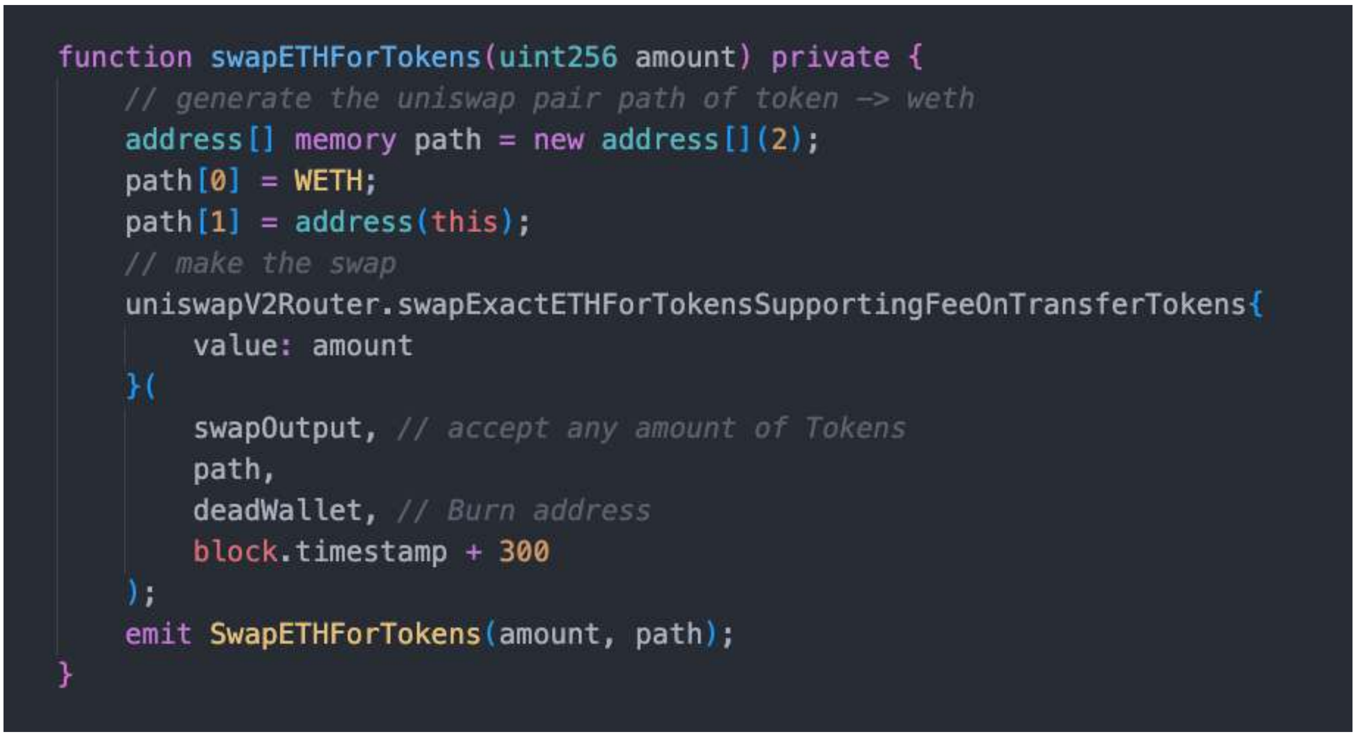
Security issue checking status

❖ **High severity issues**
No High severity issues found

❖ **Medium severity issues**
No medium severity issues found

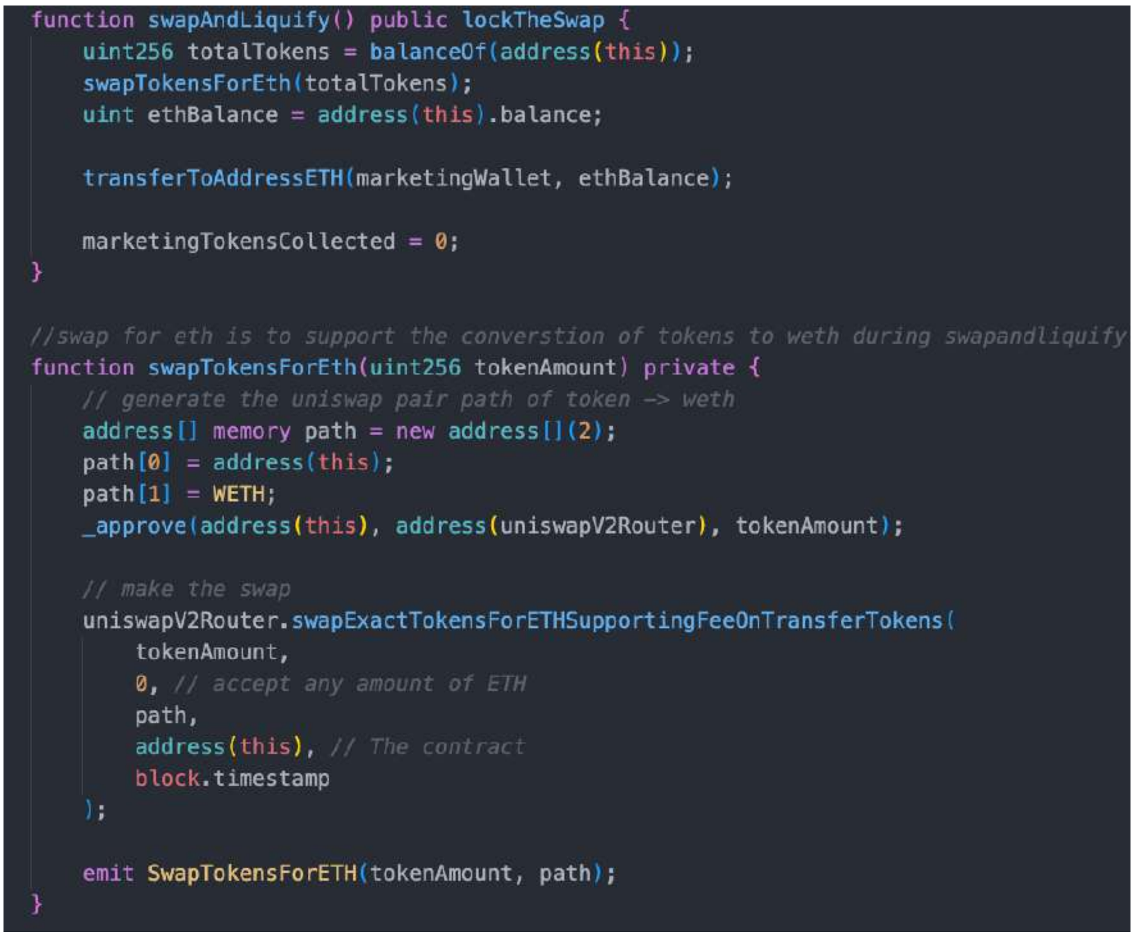
❖ **Low severity issues**

The "swapbnbForTokens" function is not utilized within the contract and does not serve any purpose in its functions. Therefore, it is advisable to remove this function from the contract as it is unnecessary.



```
function swapETHForTokens(uint256 amount) private {
    // generate the uniswap pair path of token -> weth
    address[] memory path = new address[](2);
    path[0] = WETH;
    path[1] = address(this);
    // make the swap
    uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{
        value: amount
    }(
        swapOutput, // accept any amount of Tokens
        path,
        deadWallet, // Burn address
        block.timestamp + 300
    );
    emit SwapETHForTokens(amount, path);
}
```

In the "swapAndLiquify" function, all the swapped BNB is sent directly to the marketing wallet without any additional calculations. Therefore, instead of swapping BNB to the contract and then transferring it to the marketing wallet, it is more efficient to directly swap the tokens to BNB using the "swapTokensForbnb" function and send the resulting BNB to the marketing wallet. This eliminates the unnecessary step of involving the contract in the swap process.

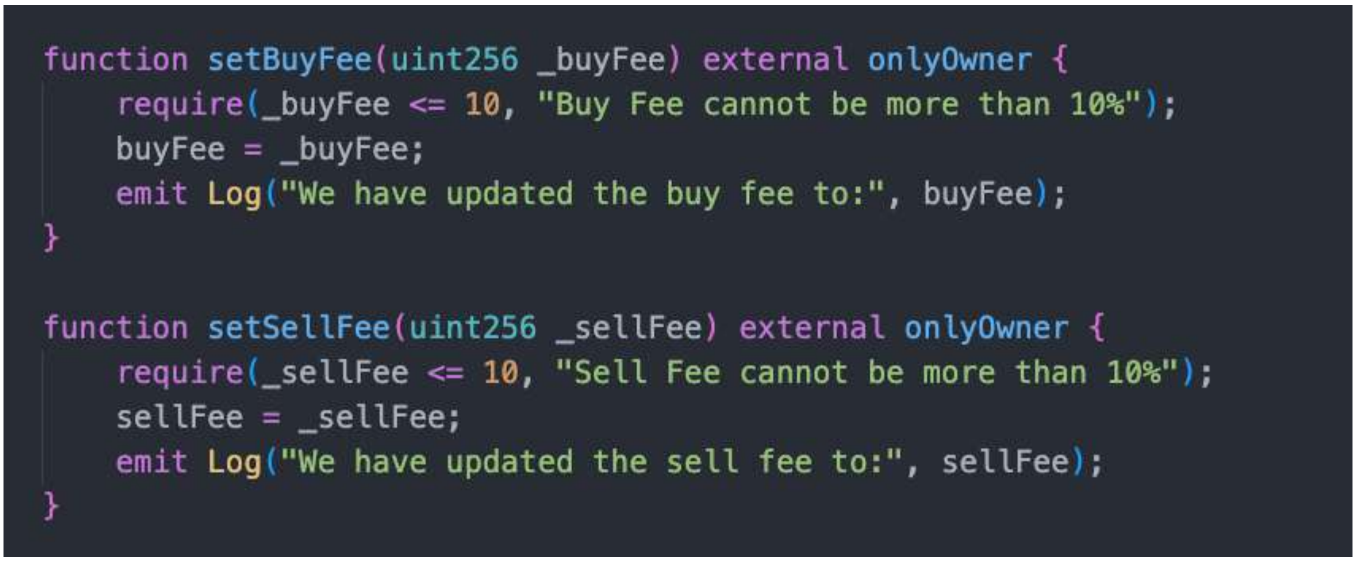


```
function swapAndLiquify() public lockTheSwap {
    uint256 totalTokens = balanceOf(address(this));
    swapTokensForEth(totalTokens);
    uint ethBalance = address(this).balance;

    transferToAddressETH(marketingWallet, ethBalance);

    marketingTokensCollected = 0;
}

//swap for eth is to support the converstion of tokens to weth during swapandliquify
function swapTokensForEth(uint256 tokenAmount) private {
    // generate the uniswap pair path of token -> weth
    address[] memory path = new address[](2);
    path[0] = address(this);
    path[1] = WETH;
    _approve(address(this), address(uniswapV2Router), tokenAmount);

    // make the swap
    uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
        tokenAmount,
        0, // accept any amount of ETH
        path,
        address(this), // The contract
        block.timestamp
    );

    emit SwapTokensForETH(tokenAmount, path);
}
```

- **Centralization Risk**
  No Centralization issues found

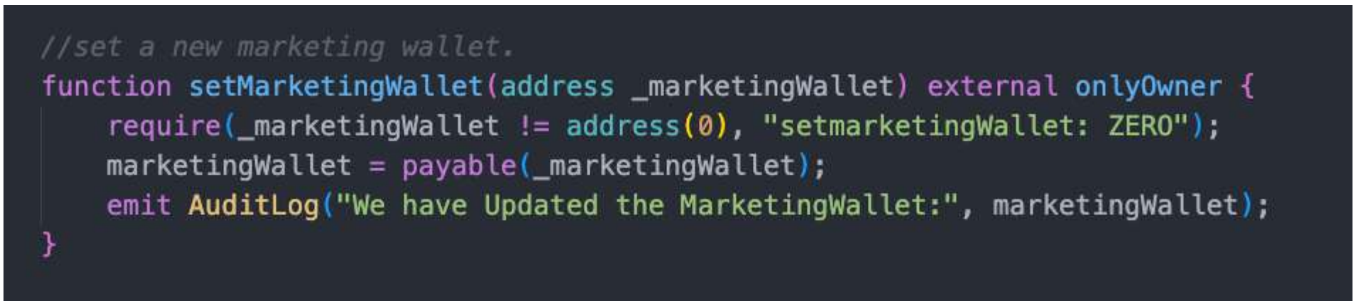
Owner privileges

- ❖ The owner can set buy and sell fees of up to 10%



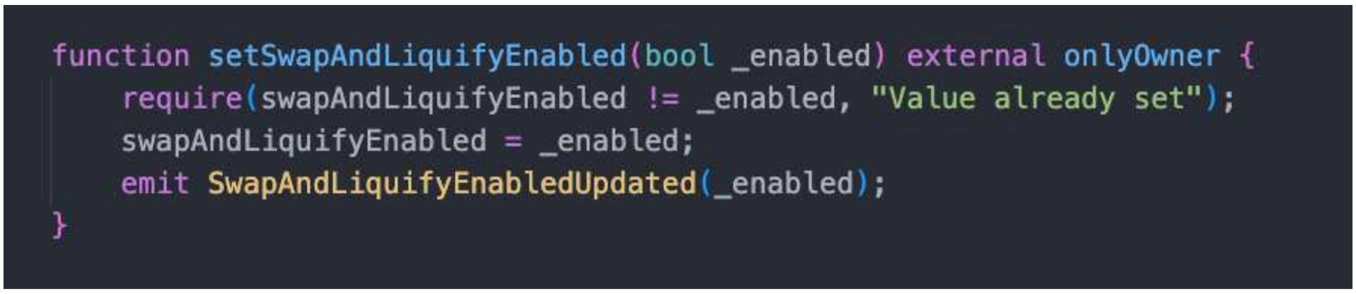
```
function setBuyFee(uint256 _buyFee) external onlyOwner {
    require(_buyFee <= 10, "Buy Fee cannot be more than 10%");
    buyFee = _buyFee;
    emit Log("We have updated the buy fee to:", buyFee);
}

function setSellFee(uint256 _sellFee) external onlyOwner {
    require(_sellFee <= 10, "Sell Fee cannot be more than 10%");
    sellFee = _sellFee;
    emit Log("We have updated the sell fee to:", sellFee);
}
```

- ❖ The owner can change the marketing wallet



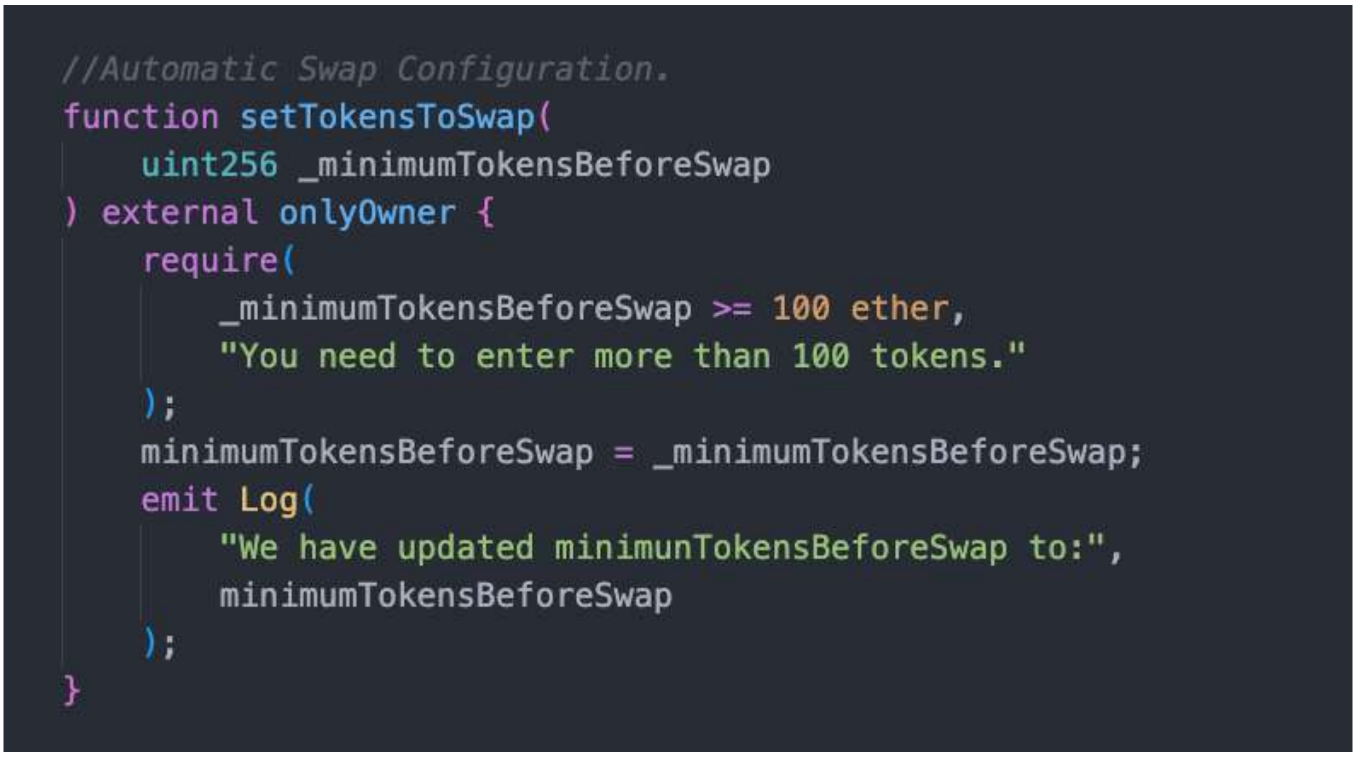
```
//set a new marketing wallet.
function setMarketingWallet(address _marketingWallet) external onlyOwner {
    require(_marketingWallet != address(0), "setmarketingWallet: ZERO");
    marketingWallet = payable(_marketingWallet);
    emit AuditLog("We have Updated the MarketingWallet:", marketingWallet);
}
```

- ❖ The owner can enable/disable swapping



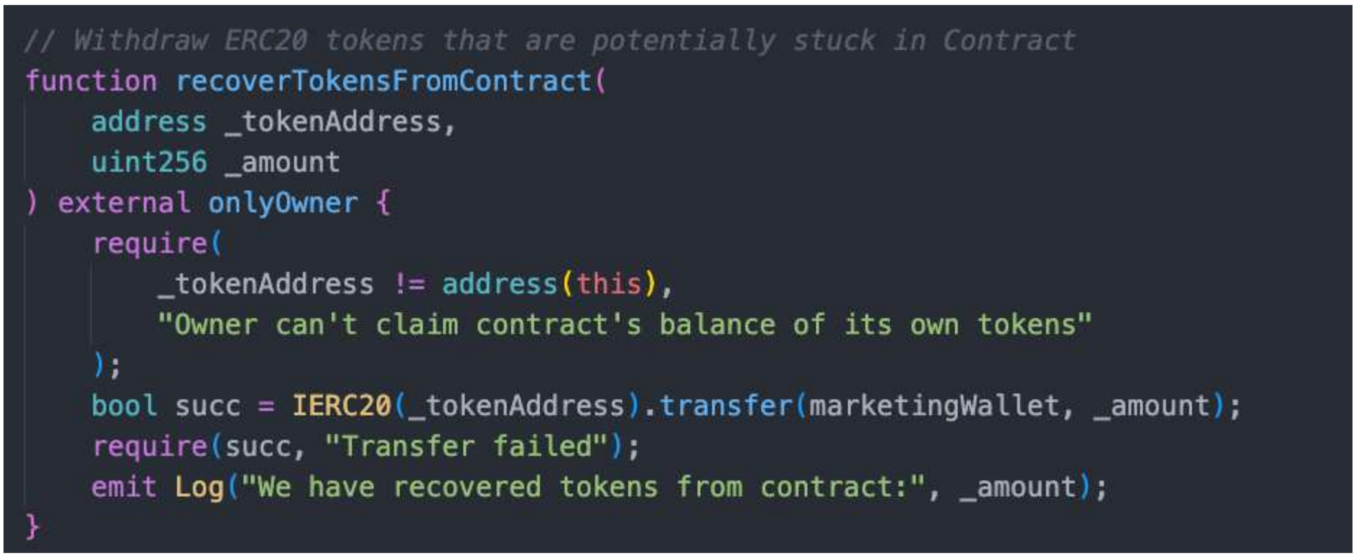
```
function setSwapAndLiquifyEnabled(bool _enabled) external onlyOwner {
    require(swapAndLiquifyEnabled != _enabled, "Value already set");
    swapAndLiquifyEnabled = _enabled;
    emit SwapAndLiquifyEnabledUpdated(_enabled);
}
```

- ❖ The owner can change the swap point minimum of upto 100 tokens



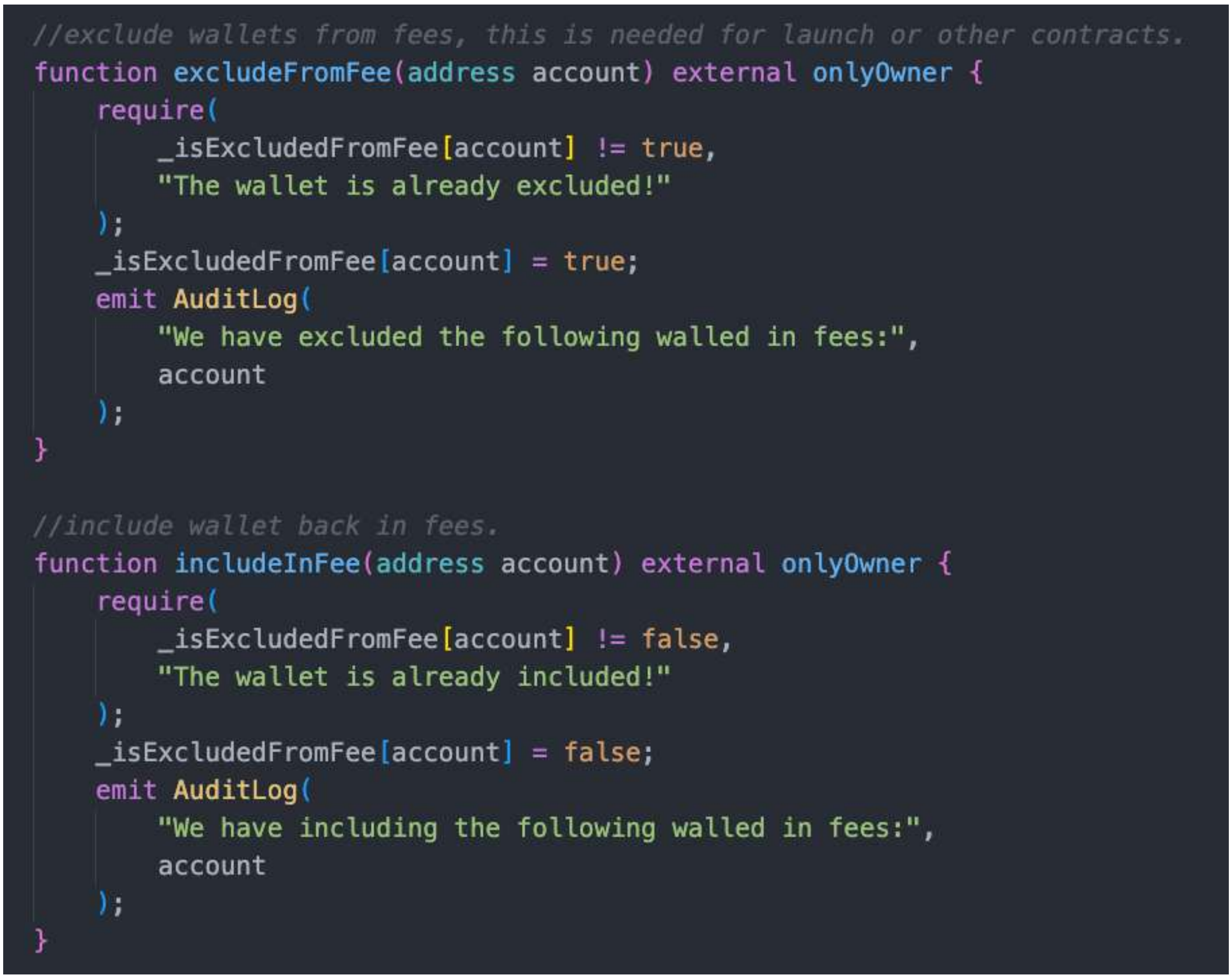
```
//Automatic Swap Configuration.
function setTokensToSwap(
    uint256 _minimumTokensBeforeSwap
) external onlyOwner {
    require(
        _minimumTokensBeforeSwap >= 100 ether,
        "You need to enter more than 100 tokens."
    );
    minimumTokensBeforeSwap = _minimumTokensBeforeSwap;
    emit Log(
        "We have updated minimunTokensBeforeSwap to:",
        minimumTokensBeforeSwap
    );
}
```

- ❖ The owner can get any BEP20 tokens from the contract to the marketing wallet (can not get native tokens)



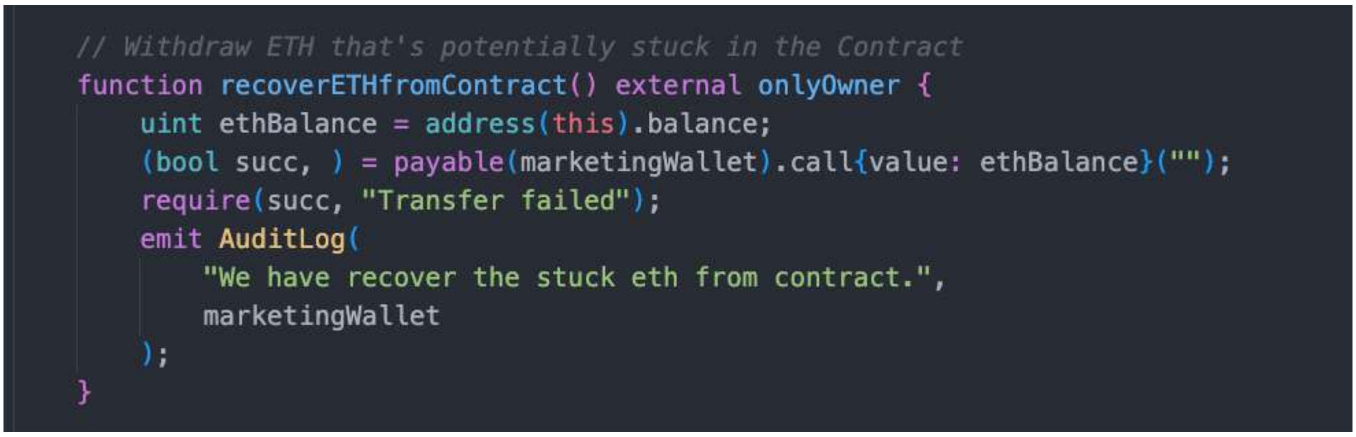
```
// Withdraw ERC20 tokens that are potentially stuck in Contract
function recoverTokensFromContract(
    address _tokenAddress,
    uint256 _amount
) external onlyOwner {
    require(
        _tokenAddress != address(this),
        "Owner can't claim contract's balance of its own tokens"
    );
    bool succ = IERC20(_tokenAddress).transfer(marketingWallet, _amount);
    require(succ, "Transfer failed");
    emit Log("We have recovered tokens from contract:", _amount);
}
```

❖ The owner can include/exclude wallets from the fees



```
//exclude wallets from fees, this is needed for launch or other contracts.
function excludeFromFee(address account) external onlyOwner {
    require(
        _isExcludedFromFee[account] != true,
        "The wallet is already excluded!"
    );
    _isExcludedFromFee[account] = true;
    emit AuditLog(
        "We have excluded the following walled in fees:",
        account
    );
}

//include wallet back in fees.
function includeInFee(address account) external onlyOwner {
    require(
        _isExcludedFromFee[account] != false,
        "The wallet is already included!"
    );
    _isExcludedFromFee[account] = false;
    emit AuditLog(
        "We have including the following walled in fees:",
        account
    );
}
```

❖ The owner can get the contract BNB balance to the marketing address



```
// Withdraw ETH that's potentially stuck in the Contract
function recoverETHfromContract() external onlyOwner {
    uint ethBalance = address(this).balance;
    (bool succ, ) = payable(marketingWallet).call{value: ethBalance}("");
    require(succ, "Transfer failed");
    emit AuditLog(
        "We have recover the stuck eth from contract.",
        marketingWallet
    );
}
```

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: **2**

Solidity code functional issue level: **PASS**

Number of owner privileges: **7**

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

Smart contract active ownership: **ACTIVE**