



RUGFREECOINS



Bot Pad Token

RugfreeCoins Verified on January 11th, 2024

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 8%.
- ✓ 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 {  
    tradingActive = true;  
    swapEnabled = true;  
    launchedAt = 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.....	17
Security issue checking status.....	18
Owner privileges.....	21
Audit conclusion.....	23

Audit details



Audited project
Bot Pad Token



Contract Address
0x5e9Af44b6e217933a4c61Dc91e9fDF490A82e2f8



Client contact
Bot Pad Token Team



Blockchain
Binance Smart chain



Project website
<https://botpad.app/>

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 **Bot Pad Token Team** to perform an audit of the smart contract.

<https://bscscan.com/token/0x5e9Af44b6e217933a4c61Dc91e9fDF490A82e2f8>

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












▲ 4% tax when buying & selling

4% of trade goes to the fee receiver in tokens.

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 Bot Pad token ecosystem.
- ▶ Anyone who's interested in taking part in the future plans of Bot Pad Token.
- ▶ Anyone who's interested in making financial transactions with any other party using Bot Pad 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 11th of January 2024







Contract name	BotPad
Contract address	0x5e9Af44b6e217933a4c61Dc91e9fDF490A82e2f8
Token supply	100,000,000
Token ticker	BPAD
Decimals	18
Token holders	1
Transaction count	1
Contract deployer address	0x1FbB30c294F318099B10288726E60FC9d563e532
Contract's current owner address	0x1FbB30c294F318099B10288726E60FC9d563e532
Fee Wallet	0x5bb2ad9C9187d3FC33a3d773EdE396D617ae5bEe

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	PASS ▾
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
Context	Implementation			
L	_msgSender	Internal 		
L	_msgData	Internal 		
IPancakeswap V2Pair	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 !
IPancakeswap V2Factory	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 !
IERC20 Metadata	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 		
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	toInt256Safe	Internal 		
SafeMathUint	Library			
L	toInt256Safe	Internal 		
IPancakeswapV2Router01	Interface			
L	factory	External !		NO !
L	WETH	External !		NO !

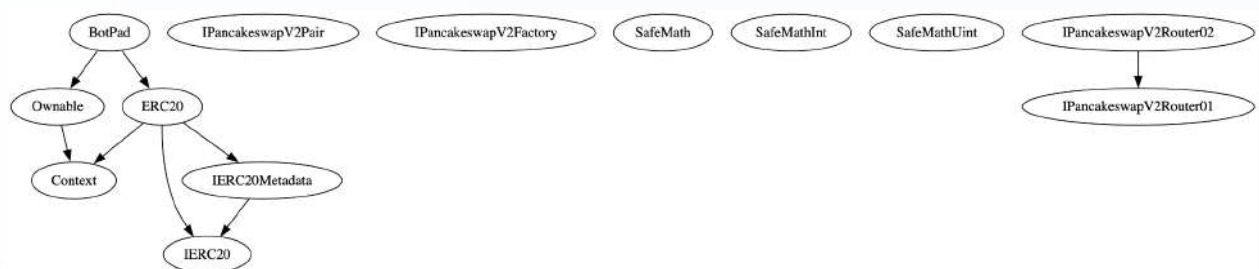
L	addLiquidity	External !		NO !
L	addLiquidityETH	External !		NO !
L	removeLiquidity	External !		NO !
L	removeLiquidityETH	External !		NO !
L	removeLiquidityWithPermit	External !		NO !
L	removeLiquidityETHWithPermit	External !		NO !
L	swapExactTokensForTokens	External !		NO !
L	swapTokensForExactTokens	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 !
IPancakeswap V2Router02	Interface	IPancakesw apV 2Router01		
L	removeLiquidityETHSupportingFeeOnTransferTokens	External !		NO !
L	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External !		NO !
L	swapExactTokensForTokensSupportingFeeOnTransferTokens	External !		NO !
L	swapExactETHForTokensSupportingFeeOnTransferTokens	External !		NO !
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External !		NO !

BotPad	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	updatefeeswallet	External !	🛑	onlyOwner
L	isExcludedFromFees	Public !		NO !
L	_transfer	Internal 🔒	🛑	

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 {  
    tradingActive = true;  
    swapEnabled = true;  
    launchedAt = block.number;  
}
```

The owner can change the max wallet limit for less than 2% (Fixed)

```
function updateMaxWalletAmount(uint256 newNum) external onlyOwner {  
    require(newNum >= (totalSupply() * 5 / 1000)/1e18, "Cannot set maxWallet  
lower than 0.5%");  
    maxWallet = newNum * (10**18);  
}
```

The fee address is hardcoded in the contract, and the owner cannot change it. The owner can only modify the marketing wallet, but the marketing wallet does not receive fees; the fee address is the actual fees-receiving wallet. (fixed)

```
address feesaddress= 0x5f793BEc452356AED90D861a3d675bd58127c1Df;
```

❖ Medium severity issues

All fees are directly sent to the fees wallet, and fees do not reach the contract. Consequently, there will be no tokens from the fees in the contract to swap. As a result, all functions related to swapping are redundant. **fixed**

```
function swapTokensForEth(uint256 tokenAmount) private {

    // generate the pancakeswap pair path of token -> weth
    address[] memory path = new address[](2);
    path[0] = address(this);
    path[1] = pancakeswapV2Router.WETH();

    _approve(address(this), address(pancakeswapV2Router), tokenAmount);

    // make the swap
    pancakeswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
        tokenAmount,
        0, // accept any amount of ETH
        path,
        address(this),
        block.timestamp
    );
}

function swapBack() private {
    uint256 contractBalance = balanceOf(address(this));
    bool success;

    if(contractBalance == 0) {return;}

    if(contractBalance > swapTokensAtAmount * 20){
        contractBalance = swapTokensAtAmount * 20;
    }

    uint256 amountToSwapForETH = contractBalance;
    swapTokensForEth(amountToSwapForETH);

    (success,) = address(marketingWallet).call{value: address(this).balance}(
        ""
    );
}
```

❖ Low severity issues

The owner can change buy and sell fees only up to 2%, but the error message incorrectly states up to 10%. The error message is incorrect. **(Fixed)**

```
function updateBuyFees(uint256 _marketingFee) external onlyOwner {
    buyMarketingFee = _marketingFee;
    buyTotalFees = buyMarketingFee;
    require(buyTotalFees <= 2, "Must keep fees at 10% or less");
}

function updateSellFees(uint256 _marketingFee) external onlyOwner {
    sellMarketingFee = _marketingFee;
    sellTotalFees = sellMarketingFee;
    require(sellTotalFees <= 2, "Must keep fees at 10% or less");
}
```

Missing visibility in fee address **(Fixed)**

```
address feesaddress= 0x5f793BEc452356AED90D861a3d675bd58127c1Df;
```

Marketing address is not used in the contract so all the functions related to the marketing address is redundant **(Fixed)**

```
address public marketingWallet;
```

Owner privileges

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

```
function enableTrading() external onlyOwner {  
    tradingActive = true;  
    swapEnabled = true;  
    launchedAt = block.number;  
}
```

- ❖ Owner can change buy and sell fees maximum upto 4% (each 2%)

```
function updateBuyFees(uint256 _marketingFee) external onlyOwner {  
    buyMarketingFee = _marketingFee;  
    buyTotalFees = buyMarketingFee;  
    require(buyTotalFees <= 2, "Must keep fees at 10% or less");  
}  
  
function updateSellFees(uint256 _marketingFee) external onlyOwner {  
    sellMarketingFee = _marketingFee;  
    sellTotalFees = sellMarketingFee;  
    require(sellTotalFees <= 2, "Must keep fees at 10% or less");  
}
```

❖ 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 new lp pairs

```
function setAutomatedMarketMakerPair(address pair, bool value) public onlyOwner  
{  
    require(pair != pancakeswapV2Pair, "The pair cannot be removed from  
automatedMarketMakerPairs");  
    _setAutomatedMarketMakerPair(pair, value);  
}
```

❖ Owner can change marketing wallet (redundant)

```
function updateMarketingWallet(address newMarketingWallet) external onlyOwner {  
    emit marketingWalletUpdated(newMarketingWallet, marketingWallet);  
    marketingWallet = newMarketingWallet;  
}
```

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:	1
Solidity code functional issue level:	PASS ▾
Number of owner privileges:	5
Centralization risk correlated to the active owner:	HIGH ▾
Smart contract active ownership:	ACTIVE ▾