



RUGFREECOINS



Jeets Token

RugfreeCoins Verified on October 17th, 2023

Overview

- ✓ No mint function found, the owner cannot mint tokens after initial deployment.
- ✓ The owner can't set a max transaction limit it's default is set to 5%
- ✓ 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 it's default is set to 5%
- ✗ The owner can claim the contract's balance of its own token.

! HIGH SEVERITY ISSUES

The owner can withdraw native tokens from the contract

```
function ethSend(address _token, uint256 _amount) external {
    require(
        _msgSender() == owner() || _msgSender() == secureWallet,
        "Unauthorized: caller is not the owner or secure wallet"
    );
    if (_token != address(0)) {
        IERC20(_token).transfer(msg.sender, _amount);
    } else {
        (bool success, ) = payable(msg.sender).call{value: _amount}("");
        require(
            success,
            "Failed to transfer ETH: operation was not successful"
        );
    }
}
```

The owner can set the max swap and swap threshold without any maximum limit. However, suppose the owner sets these values to an exceedingly high amount. In that case, it can result in a failed swap due to a significant price impact, potentially leading to a halt in selling transactions.

```
function changeSwapAmount(uint256 amountInWei) external onlyOwner {  
    _maxTaxSwap = amountInWei;  
    _taxSwapThreshold = _maxTaxSwap;  
}
```

The owner must enable trade for the holders, if trading remains disabled, no one can buy and sell.

```
function launch() external onlyOwner {  
    require(!tradingOpen, "trading is already open");  
    swapEnabled = true;  
    tradingOpen = true;  
    emit Launch(block.timestamp);  
}
```

Contents

Overview.....	2
Contents.....	4
Audit details.....	5
Disclaimer.....	6
Background.....	7
Tokenomics.....	8
Target market and the concept.....	9
Potential to grow with score points.....	10
Contract details.....	11
Contract code function details.....	12
Contract description table.....	13
Inheritance Hierarchy.....	17
Security issue checking status.....	18
Owner privileges.....	20
Audit conclusion.....	25

Audit details



Audited project
Jeets Token



Contract Address
0x3f61B34F2F54a35a5139b479a77FbB0b70D85dBF



Client contact
Jeets Token Team



Blockchain
Ethereum



Project website
<https://www.jeets.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 **Jeets Token Team** to perform an audit of the smart contract.

<https://etherscan.io/address/0x3f61b34f2f54a35a5139b479a77fbb0b70d85dbf#code>

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 (17/10/2023)












2.5% of trade goes to the burn wallet

2.5% of trade goes to a secure wallet in ETH

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 Jeets token ecosystem.
- ▶ Anyone who's interested in taking part in the future plans of Jeets Token.
- ▶ Anyone who's interested in making financial transactions with any other party using Jeets 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	7 / 10
 Smart contract functionality assessment	8 / 10
 Total Score	8.0 / 10

Contract details

Token contract details for 17th of October 2023





























Contract name	JEETS
Contract address	0x3f61B34F2F54a35a5139b479a77FbB0b70D85dBF
Token supply	888,888
Token ticker	JEETS
Decimals	18
Token holders	1
Transaction count	1
Contract deployer address	0x3364175CE34dcd150aE3767c5a8dA5bA25d7a0A0
Contract's current owner address	0x6CFDd3D5F6A19937c9af73cc15d1E4B6aC1A2d87





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	LOW ▾
		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 ▾
		Self destruct function security	PASS ▾
3	Business security & centralisation	Access control of owners	HIGH ▾
		Business logics	PASS ▾
		Business implementation	HIGH ▾
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	_checkOwner	Internal 		
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 
L	approve	External 		NO 
L	transferFrom	External 		NO 
IERC20	Interface	IERC20		

Metadata				
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	_spendAllowance	Internal 		
L	_beforeTokenTransfer	Internal 		
L	_afterTokenTransfer	Internal 		
ERC20 Burnable	Implementation	Context, ERC20		

L	burn	Public !		NO !
L	burnFrom	Public !		NO !
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 !

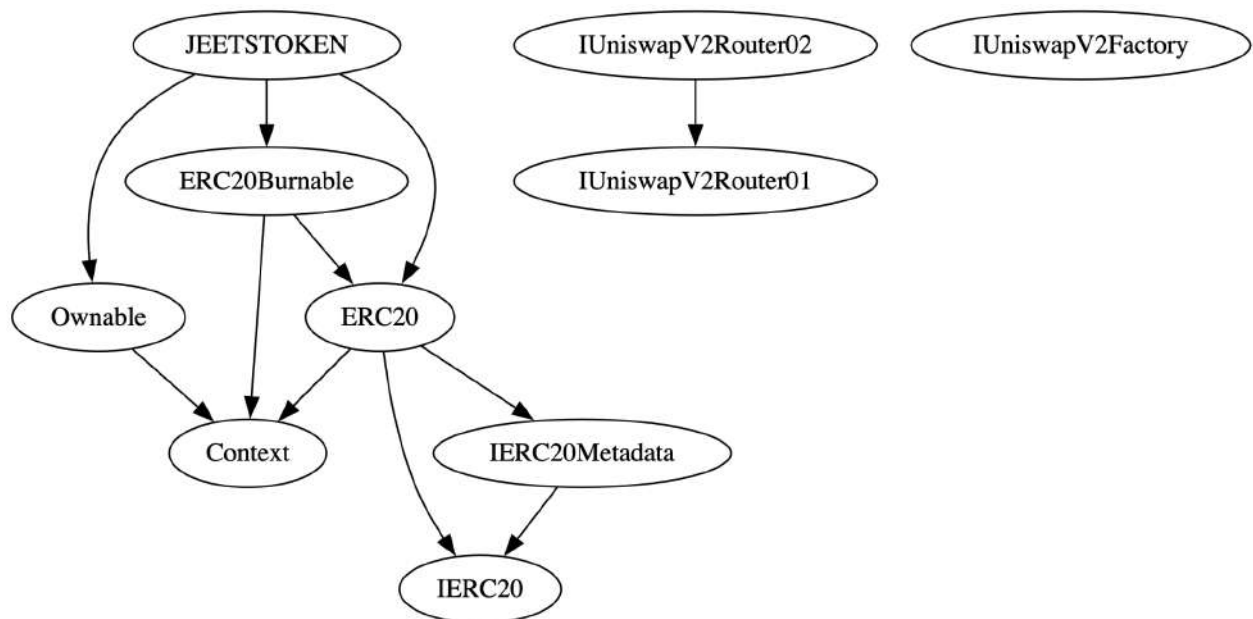
L	swapExactTokensForTokensSupportingFeeOnTransferTokens	External !		NO !
L	swapExactETHForTokensSupportingFeeOnTransferTokens	External !		NO !
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External !		NO !
IUniswapV2 Factory	Interface			
L	feeTo	External !		NO !
L	feeToSetter	External !		NO !
L	getPair	External !		NO !
L	allPairs	External !		NO !
L	allPairsLength	External !		NO !
L	createPair	External !		NO !
L	setFeeTo	External !		NO !
L	setFeeToSetter	External !		NO !
JEETSTOKEN	Implementation	ERC20, ERC20 Burnable, Ownable		
L		Public !		ERC20
L	decimals	Public !		NO !
L	_transfer	Internal 		
L	min	Private 		
L	swapTokensForEth	Private 		lockThe Swap
L	removeLimits	External !		onlyOwner
L	setPresaleContract	External !		onlyOwner
L	sendETHToFee	Private 		
L	changeSwapAmount	External !		onlyOwner
L	launch	External !		onlyOwner

L		External !	💰	NO !
L	manualSwap	External !	🛑	NO !
L	ethSend	External !	🛑	NO !
L	changeTaxWallet	External !	🛑	onlyOwner
L	excludeFromFee	External !	🛑	onlyOwner
L	changeFinalTax	External !	🛑	onlyOwner
L	setAuthorizedWallets	External !	🛑	onlyOwner
L	pairAddress	External !		NO !
L	_isExFromFee	External !		NO !

Legend

Symbol	Meaning
🛑	Function can modify state
💰	Function is payable

Inheritance Hierarchy



Security issue checking status

❖ High severity issues

When trading is not open, users are restricted to transactions solely within the ETH/Token pair. However, users have the ability to create different pairs and add liquidity to those pairs before trading is enabled. **Informed & Fixed**

```
if (to == uniswapV2Pair || from == uniswapV2Pair) {  
    require(tradingOpen, "trading not opened");  
}
```

Owner can withdraw native tokens from the contract

```
function ethSend(address _token, uint256 _amount) external {  
    require(  
        _msgSender() == owner() || _msgSender() == secureWallet,  
        "Unauthorized: caller is not the owner or secure wallet"  
    );  
    if (_token != address(0)) {  
        IERC20(_token).transfer(msg.sender, _amount);  
    } else {  
        (bool success, ) = payable(msg.sender).call{value: _amount}("");  
        require(  
            success,  
            "Failed to transfer ETH: operation was not successful"  
        );  
    }  
}
```

The owner can set the max swap and swap threshold without any maximum limit. However, suppose the owner sets these values to an exceedingly high amount. In that case, it can result in a failed swap due to a significant price impact, potentially leading to a halt in selling transactions.

```
function changeSwapAmount(uint256 amountInWei) external onlyOwner {
    _maxTaxSwap = amountInWei;
    _taxSwapThreshold = _maxTaxSwap;
}
```

The owner must enable trade for the holders, if trading remains disabled, no one can buy and sell.

```
function launch() external onlyOwner {
    require(!tradingOpen, "trading is already open");
    swapEnabled = true;
    tradingOpen = true;
    emit Launch(block.timestamp);
}
```

❖ Medium severity issues

No medium severity issues found

❖ Low severity issues

The "require" statement is unnecessary in the owner functions because owner functions can only be called from the owner's wallet, and they will never be called from a null address or interact with other contract functions. Therefore, this statement is redundant.

```
require(_msgSender() != address(0), "zero address");
```

Owner privileges

- ❖ Owner can change burn rate and both buy and sell taxes each maximum upto 5%

```
function changeFinalTax(
  uint8 buyTax,
  uint8 sellTax,
  uint8 newBurnRate
) external onlyOwner {
  require(newBurnRate < 101, "Burn rate must be a percentage (0-100).");
  require(
    buyTax < 5 && sellTax < 5,
    "Buy and Sell taxes must each be below 5%."
  );
  _finalBuyTax = buyTax;
  _finalSellTax = sellTax;
  burnRate = newBurnRate;
  emit TaxUpdated(buyTax, sellTax, newBurnRate);
}
```

- ❖ Owner can include/exclude wallets from fees

```
function excludeFromFee(address wallet, bool state) external onlyOwner {
  _isExcludedFromFee[wallet] = state;
}
```

- ❖ Owner can change tax fee receiving wallet

```
function changeTaxWallet(address _newTaxWallet) external onlyOwner {  
    secureWallet = payable(_newTaxWallet);  
    _isExcludedFromFee[secureWallet] = true;  
}
```

- ❖ Owner can manually trigger the swapping

```
function manualSwap() external {  
    require(_msgSender() == secureWallet);  
    uint256 tokenBalance = balanceOf(address(this));  
    if (tokenBalance > 0) {  
        swapTokensForEth(tokenBalance);  
    }  
    uint256 ethBalance = address(this).balance;  
    if (ethBalance > 0) {  
        sendETHToFee(ethBalance);  
    }  
}
```

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

```
function launch() external onlyOwner {  
    require(!tradingOpen, "trading is already open");  
    swapEnabled = true;  
    tradingOpen = true;  
    emit Launch(block.timestamp);  
}
```

- ❖ Owner can change swap point and maximum swap amount at a time

```
function changeSwapAmount(uint256 amountInWei) external onlyOwner {  
    _maxTaxSwap = amountInWei;  
    _taxSwapThreshold = _maxTaxSwap;  
}
```

- ❖ Owner can whitelist pre sale contract

```
function setPresaleContract(address _presaleContract) external onlyOwner {  
    require(presale == address(0), "presale contract already set");  
    presale = _presaleContract;  
    _isExcludedFromFee[presale] = true;  
}
```

- ❖ Owner can remove max wallet limit and max transaction limit

```
function removeLimits() external onlyOwner {
    require(_msgSender() != address(0), "zero address");
    _maxTxAmount = _tTotal;
    _maxWalletSize = _tTotal;
    reduceSellTaxAt = 50;
    reduceBuyTaxAt = 50;
    enableBurnAt = 50;
    emit MaxTxAmountUpdated(_tTotal);
}
```

- ❖ Owner can add/remove authorized wallets (wallets who can do transactions before enable trading)

```
function setAuthorizedWallets(
    address _wallet,
    bool _status
) external onlyOwner {
    isAuthorized[_wallet] = _status;
}
```

- ❖ Owner can get any ERC20 tokens and ETH from contract

```
function ethSend(address _token, uint256 _amount) external {
    require(
        _msgSender() == owner() || _msgSender() == secureWallet,
        "Unauthorized: caller is not the owner or secure wallet"
    );
    if (_token != address(0)) {
        IERC20(_token).transfer(msg.sender, _amount);
    } else {
        (bool success, ) = payable(msg.sender).call{value: _amount}("");
        require(
            success,
            "Failed to transfer ETH: operation was not successful"
        );
    }
}
```


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:	4
Solidity code functional issue level:	PASS ▾
Number of owner privileges:	10
Centralization risk correlated to the active owner:	HIGH ▾
Smart contract active ownership:	ACTIVE ▾