



RugFreeCoins Audit



Naruto 2.0 Token Smart Contract Security Audit

July 15th ,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.
- ✓ 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

Overview	ii
Audit details	1
Disclaimer	2
Background	3
Roadmap	4
Target market and the concept	6
Potential to grow with score points	7
Total Points	7
Contract details	8
Contract code function details	9
Contract description table	11
Security issue checking status	17
Owner privileges	19
Audit conclusion	20

Audit details



Audited project

Naruto 2.0 Token



Contract Address

0xda2378A5E7005e83e24557F4D74D5F52A2Cf8abF



Client contact

Naruto 2.0 Token Team



Blockchain

Binance Smart chain



Project website

<https://naruto2-0.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 Naruto 2.0 Token Team to perform an audit of the smart contract.

<https://bscscan.com/address/0xda2378A5E7005e83e24557F4D74D5F52A2Cf8abF>

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

Phase 01

- Naruto 2.0 idea conception
- Build website
- Build tokenomics
- Deploy smart contract Naruto 2.0
- More than 650 members in the community
- Audit

Phase 02

- Small marketing
- Community Competitions
- 4500 Community Members
- FairLaunch in Pinksale
- Launch Naruto 2.0 in PancakeSwap
- Trending AVE, DEXView, Dextools.
- Big marketing with influencer
- Calls Tier 1
- Coin Market Cap (CMC) fast tracking

Phase 03

- Listing in CEX top 10
- Merchandising with logo Naruto 2.0
- HOTBIT, MEXC listing
- Burnt
- 10000 holders
- Big campaign marketing

Phase 04

- Top 100 CMC
- Positioning for listing on Binance and Kucoin

Tokenomics (15/01/2023)

5% tax when buying

- 1% trade distributes among holders as rewards in tokens.
- 4% of trade goes to the marketing wallets in BNB

3% tax when selling

- 1% trade distributes among holders as rewards in tokens.
- 2% of trade goes to the marketing wallets 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 Naruto 2.0 token ecosystem.
- Anyone who's interested in taking part in the future plans of Naruto 2.0 Token.
- Anyone who's interested in making financial transactions with any other party using Naruto 2.0 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	8/10
8	Preparing for the future	8/10
9	Smart contract security	10/10
10	Smart contract functionality assessment	9/10
Total Points		8.3/10

Contract details

Token contract details for 15th of July 2023

Contract name	Naruto 2.0
Contract address	0xda2378A5E7005e83e24557F4D74D5F52A2Cf8abF
Token supply	100,100,000,000
Token ticker	Naruto 2.0
Decimals	9
Token holders	1
Transaction count	1
Contract deployer address	0x5C08AFB923630dd61959FF779431ed0438D0C34C
Contract's current owner address	0x5c08afb923630dd61959ff779431ed0438d0c34c






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
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 !
Context	Implementation			
L	_msgSender	Internal 		
L	_msgData	Internal 		
IERC20 Metadata	Interface	IERC20		
L	name	External !		NO !
L	symbol	External !		NO !
L	decimals	External !		NO !



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 🔒		
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 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	Iuniswap V2 Router01		

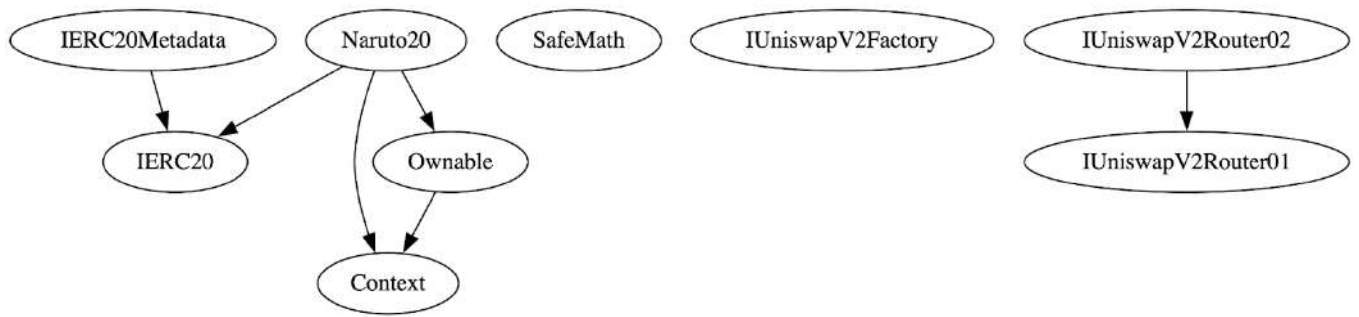
L	removeLiquidityETHSupportingFeeOnTransferTokens	External !		NO !
L	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External !		NO !
L	swapExactTokensForTokensSupportingFeeOnTransferTokens	External !		NO !
L	swapExactETHForTokensSupportingFeeOnTransferTokens	External !		NO !
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External !		NO !
Ownable	Implementation	Context		
L		Public !		NO !
L	owner	Public !		NO !
L	renounceOwnership	Public !		onlyOwner
Naruto20	Implementation	Context, IERC20, Ownable		
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	tokenFromReflection	Private 		
L	_approve	Private 		
L	_transfer	Private 		
L	swapTokensForEth	Private 		lockThe Swap
L	sendETHToFee	Private 		
L	_tokenTransfer	Private 		
L	_transferStandard	Private 		
L	_reflectFee	Private 		
L		External !		NO !
L	_getRate	Private 		
L	_getCurrentSupply	Private 		
L	manualswap	External !		NO !
L	manualsend	External !		NO !
L	excludeMultipleAccountsFromFees	Public !		onlyOwner
L	excludeFromFees	Public !		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

In the transfer function, there are multiple if conditions being checked in the area shown in the screenshot. However, currently, all the if conditions are being checked even though only one condition will be executed at a time during token transfer. This approach increases the gas consumption unnecessarily. To optimize this process, it would be better to utilize if-else statements instead. By incorporating if-else statements, the code can be structured in a way that prevents unnecessary evaluation of subsequent conditions once a matching condition is found. This will effectively reduce gas consumption during token transfers.

```
if (from != owner() && to != owner()) {
    uint256 contractTokenBalance = balanceOf(address(this));

    bool isJustTransfer = false;

    if (
        from != address(uniswapV2Pair) && to != address(uniswapV2Pair)
    ) {
        isJustTransfer = true;
    }

    if (
        !isJustTransfer &&
        !inSwap &&
        from != uniswapV2Pair &&
        swapEnabled &&
        contractTokenBalance > 0
    ) {
        swapTokensForEth(contractTokenBalance);
        uint256 contractETHBalance = address(this).balance;
        if (contractETHBalance > 0) {
            sendETHToFee(address(this).balance);
        }
    }

    if (from == uniswapV2Pair && to != address(uniswapV2Router)) {
        _redisFee = _redisFeeOnBuy;
        _taxFee = _taxFeeOnBuy;
    }

    if (to == uniswapV2Pair && from != address(uniswapV2Router)) {
        _redisFee = _redisFeeOnSell;
        _taxFee = _taxFeeOnSell;
    }

    if (isJustTransfer) {
        _redisFee = 0;
        _taxFee = 0;
    }

    if (_isExcludedFromFee[from] || _isExcludedFromFee[to]) {
        _redisFee = 0;
        _taxFee = 0;
    }
}
```

❖ Centralization Risk

No Centralization issues found

Owner privileges

- ❖ Owner can trigger the swap manually

```
function manualswap() external {  
    require(_msgSender() == owner());  
    uint256 contractBalance = balanceOf(address(this));  
    swapTokensForEth(contractBalance);  
}
```

- ❖ Owner can manually send contract bnb across the marketing wallets

```
function manualsend() external {  
    require(_msgSender() == owner());  
    uint256 contractETHBalance = address(this).balance;  
    sendETHToFee(contractETHBalance);  
}
```

- ❖ Owner can include/exclude wallets from fees

```
function excludeMultipleAccountsFromFees(  
    address[] calldata accounts,  
    bool excluded  
) public onlyOwner {  
    for (uint256 i = 0; i < accounts.length; i++) {  
        _isExcludedFromFee[accounts[i]] = excluded;  
    }  
    emit ExcludeMultipleAccountsFromFees(accounts, excluded);  
}  
  
event ExcludeFromFees(address indexed account, bool isExcluded);  
  
function excludeFromFees(address account, bool excluded) public onlyOwner {  
    _isExcludedFromFee[account] = excluded;  
    emit ExcludeFromFees(account, excluded);  
}
```

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: **1**

Solidity code functional issue level: **PASS**

Number of owner privileges: **3**

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

Smart contract active ownership: **ACTIVE**