



RugFreeCoins Audit



Meta Ruffy Token

Smart Contract Security Audit

January 16, 2022

Contents

Audit details	1
Disclaimer	2
Background	3
About the project	4
Target market and the concept	9
Potential to grow with score points	15
Total Points	15
Contract details	16
Token distribution	17
Contract code function details	18
Contract description table	19
Security issue checking status	28
Owner privileges	29
Audit conclusion	34

Audit details



Audited project

Meta Ruffy Token



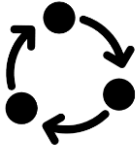
Contract Address

0x3e6227fd0e67fe830fb274d0b11845742ef336e0



Client contact

Meta Ruffy Team



Blockchain

Binance smart chain



Project website

<https://metaruffy.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 Meta Ruffy Token to perform an audit of the smart contract.

<https://bscscan.com/address/0x3e6227fd0e67fe830fb274d0b11845742ef336e0>

The focus of this audit is to verify 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, long term sustainability and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

About the project

Meta Ruffy is a token built on the Binance Smart Chain that is with an innovative investment use case the main purpose of which is to seek out constant revenue sources, and heading towards building even greater Community, with combining latest technology which merges together, web3.0, blockchain, VR and AR into their business to dominate the entertainment area in the Metaverse. Each transaction, purchase incurs a 10% fee, and sale incurs a 14% fee.

Features

- ❖ The **Meta Ruffy rewards BUSD** will be distributed among every holder proportional to how many tokens each individual holds in values of **5% when buying and 6% when selling**.
- ❖ The **sustainability fee of 3% development when buying and selling and 3% for marketing when selling** is what allows Meta Ruffy to hold the aforementioned promise. Tokens will be swapped into BNB and will be sent to a marketing wallet per transaction. This way, Meta Ruffy will have enough funds to promote the coin and spend for future development without selling tokens as the traditional way.
- ❖ Meta Ruffy has the burn strategy that benefits and rewards those who invest long-term. A **2% fee will be charged when buying and selling** and sent to the burn wallet. This feature slowly reduces supply making each Meta Ruffy more and more valuable.

Tokenomics

10% fee when buying

- ❖ 5% of trade goes to holders' pockets in BUSD
- ❖ 3% of trade goes to the development wallet in BNB.
- ❖ 2% of trade goes to the burn wallet.

14% fee when selling

- ❖ 6% of trade goes to holders' pockets in BUSD.
- ❖ 3% of trade goes to the development wallet in BNB.
- ❖ 3% of trade goes to the marketing wallet in BNB.
- ❖ 2% of trade goes to the burn wallet.

Roadmap

Phase 1 before privatesale

Base

- ❖ Concept design
- ❖ Create an international Name
- ❖ Website finalize Design
- ❖ Website integration final Design
- ❖ Whitepaper V2
- ❖ Roadmap V2
- ❖ Social Media Channels
- ❖ Open International groups on Telegram
- ❖ Building public Community
- ❖ Building VIP community (private group)

Token

- ❖ Deploy META RUFFY (MR) on BSC)
- ❖ Token Audit „RugFreeCoins“
- ❖ KYC (Pinksale)
- ❖ Explainer video TOKENOMICS

Ruffy world

- ❖ Start development of RUFFY WORLD v1.0
- ❖ Launch play test of RUFFY WORLD v1.0
- ❖ Last game testing (android)
- ❖ Last game testing (iOs)
- ❖ Last game testing (windows)
- ❖ Last game testing (webGL, browser)
- ❖ Last game testing (Oculus)
- ❖ Environment last polishing
- ❖ Integrate final Environment to Game server
- ❖ Explainer video RUFFY WORLD

dAPP

- ❖ Start development of RUFFY DAPP v1.0
- ❖ Implementing functionality over 20 contracts
 - Privatsale
- ❖ Staking (standard)
- ❖ Staking (mystery)
- ❖ NFT Marketplace
 - Mint your NFT
- ❖ Buy random NFT
- ❖ Design „NFT Collection 1“= 10.000pcs
- ❖ Minting „NFT Collection 1“= 10.000pcs
- ❖ NFT Wallet
- ❖ Redesign dApp
- ❖ Ruffy World soft Launch
- ❖ Designing „special NFT Collection“10pcs
- ❖ Wheel Giveaway in VIP group
- ❖ Launching Privatsale (150 BNB)

Phase 2

- ❖ Launching „buy random NFT)
- ❖ Wheel Giveaway in the official group
- ❖ Certik Audit
- ❖ Creating Presale Link
- ❖ PooCoin Advertising
- ❖ AMA on investor groups
- ❖ Pin Postings on investor groups
- ❖ START PRESALE (850 BNB)
- ❖ Final Launch on Pancakeswap - Launching Staking
- ❖ Dextools Advertising

Phase 3 - after launch

- ❖ Launching NFT Marketplace
- ❖ CoinMarketCap listing
- ❖ CoinGecko listing
- ❖ HOTBIT listing
- ❖ XT listing
- ❖ ZT listing
- ❖ LBANK listing
- ❖ Bitmart listing
- ❖ Gate.io listing

Ruffy world

- ❖ Launch Ruffy Privat Room
- ❖ Launch Ruffy Cinema
- ❖ Launch Ruffy ART Gallery
- ❖ Launch Ruffy Merchandise
- ❖ Launch Ruffy Club
- ❖ Launch Ruffy Sport's BAR
- ❖ Launch Ruffy Fitness Island
- ❖ Launch Ruffy YOGA Island
- ❖ Launch Ruffy Mall
- ❖ Launch Ruffy Landsale
- ❖ Launch Ruffy Stadium

The team

TEAM

CIHAN
CEO, FOUNDER

FOX
COO, OPERATING

SAFI
CTO, IT

ANNA MARIA
CCO, CREATIVE

FINN
CMO, CO FOUNDER

DIGITAL
HEAD OF GUERILLA MARKETING

REK_IT_RALPH
HEAD OF GUERILLA MARKETING

ANDRE
HEAD OF SOCIAL MEDIA

SR
GRAPHIC DESIGNER

KENPACHI
EXTERNAL ADVISOR

FRIEND OF CRYPTO
LEAD ADVISOR

VOLODYMYR
BLOCKCHAIN DEVELOPER

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 collecting NFTs or trading NFTs.
- ❖ Anyone who's interested in taking part with the Ruffy world concept, where users can use Entertainment Stadium, Spa & Resorts, Pub's & Party, Social-Gaming, Dating, and NFT Marketplace.
- ❖ Anyone who's interested in taking part in the p2e games.
- ❖ Anyone who's interested in staking and getting rewards.
- ❖ Anyone who's interested in taking part with the future plans of the Meta Ruffy token.
- ❖ Anyone who's interested in making financial transactions with any other party using BUSD and Meta Ruffy as the currency.

Core concept

Meta Ruffy is an open world in the Metaverse based on the latest technology which merges together, web3.0, blockchain, VR, and AR that they have made it their business to dominate the entertainment area in the Metaverse that means they are building an open world „RUFFY World“with different subject areas.

Spa resorts relaxing zones



We build luxurious hotel complexes directly on the beach with a sunbathing area to hang out and enjoy cool drinks. Of course, no SPA and YOGA area should be missing in the hotel resort. End the evening with relaxing music and a drink or calm yourself by participating in one of our yoga classes.

Yoga or fitness alone is not fun!

We will hire real yoga and fitness trainers who will do the daily exercises with you in different groups and languages.

real trainers !! no bots no videos.

Entertainment stadium



we build virtual stadiums to host concerts, comedy and theater performances. Tickets are sold through our own booking agency. The grand finale with which we will reach the mainstream is to book a world star celebrity like Eminem, Justin Timberlake, drake, Billie Elish or Ariana Grande for a concert in our stadium.

But also artists who can no longer play concerts like 2Pac, Biggie, Michael Jackson or Elvis Presley could be brought back to life thanks to artificial intelligence, we can read out past videos with artificial intelligence and create whole new concerts with a hologram that could not be distinguished from the original.

Of course, we can also host comedy shows or record our own TV show in our stadium. Tickets for each event will be NFT's. That means you can buy tickets for you, your friends or family and send it to them.

We could plan concerts or events with over 1 million people at the same time without worrying about security and logistics permits.

Pub's, bars and clubbing-areas



We build small pubs so that friends can meet and watch sports together (NFL, NBA, Champions League...) We build gigantic clubs where live DJs will perform with different genres like Hip-hop, pop, rock, rap and house music.

In our pubs in bars we are planning cooperation with sports streaming platforms such as DAZN, sky +, NFL, NBA we will build different bars for different sports.

If you want to watch American football you go to the bar where you can find like-minded people. If you prefer to watch European football, you can go to the pub and watch your favorite team win together with football fans from all over the world.

Social gaming



We are going to build a gigantic game center with all kinds of machines like pinball, car racing, motorcycle racing, basketball, billiards and all kinds of arcade games that are known from the real world with a touch of the metaverse.

This is also a great opportunity for cooperation with all kinds of Play to Earn games or other game tokens.

Dating



We are building a cute little island called „RUFFY’S love Island “, where singles can meet, exchange ideas and get to know each other. And if you need some privacy with your loved one you can rent one of our many water bungalows.

NFT marketplace



We call it RUFFY MALL Because it will be a real shopping experience. Imagine meeting up with your friends and walking through our virtual mall seeing hundreds of different local shops from all kinds of categories.

- ❖ NFT artworks
- ❖ Wearable NFT sunglasses, shoes, t-shirts, and much more
- ❖ Services: audit, design, or all kinds of freelance activities can be offered

You can even create your own Shop! How do you do that? Very easily! If you think you are a great t-shirt designer you can create and offer these in your local store, use our NFT Builder for free, and sell your own designed NFT's.

Users can put on your T-shirt and try it on, if they want to take it with them, they have the option of buying the NFT directly from you.

The Meta Ruffy reward system

5% of each transaction when buying and 6% when selling gets sent amongst all holders in BUSD rewards. The holders will be eligible to receive BUSD, every one hour, and rewards are proportional to how many tokens each individual holds.

Sustainable mechanism

The **sustainability fee of 3% when buying and selling for dev and 3% when selling for marketing** is what allows Meta Ruffy to promote the token and use funds to further the development of the platform. Tokens will be swapped into BNB and will be sent to a marketing wallet per and dev wallet. This way, Meta Ruffy will have access to the funds without selling tokens as the traditional way, which will enable them to consume funds without hurting the project.

Meta Ruffy has the burn strategy that benefits and rewards those who invest long-term. This feature slowly reduces supply making Meta Ruffy's price more and more valuable.

Potential to grow with score points

1.	Project efficiency	10/10
2.	Project uniqueness	10/10
3	Information quality	9/10
4	Service quality	9/10
5	System quality	9/10
6	Impact on the community	10/10
7	Impact on the business	10/10
8	Preparing for the future	10/10
Total Points		9.625/10

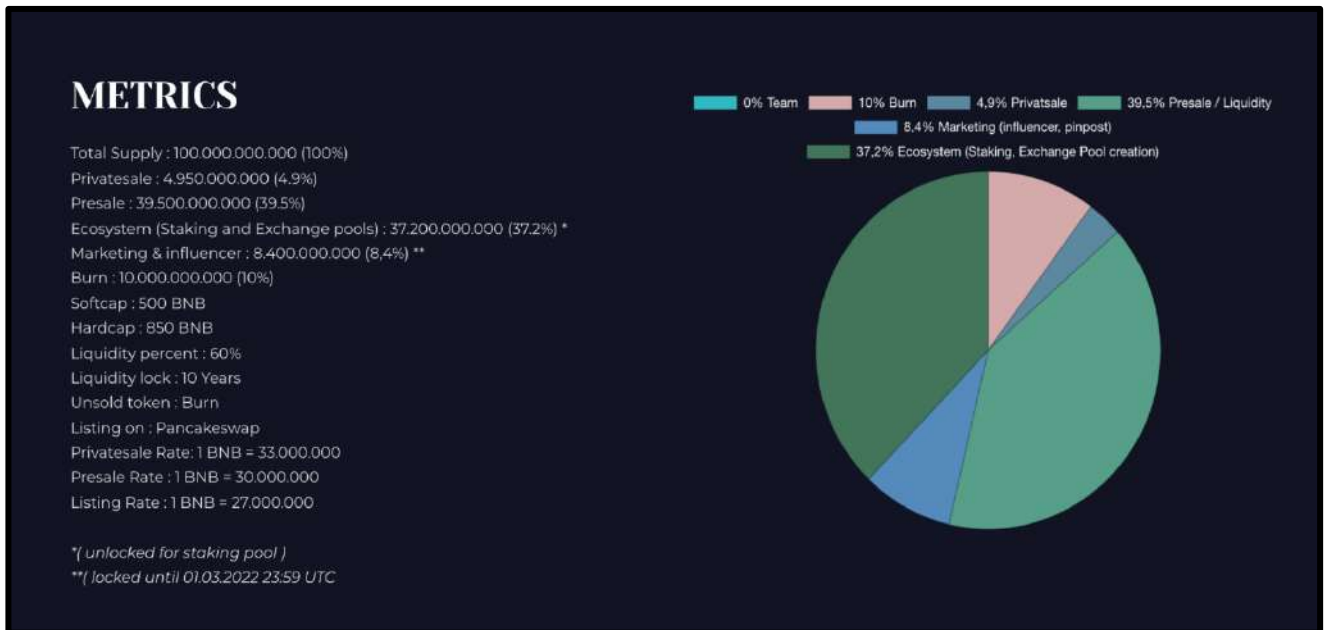
Contract details

Token contract details for 16th January 2022

Contract name	Meta Ruffy
Contract address	0x3e6227fd0e67fe830fb274d0b11845742ef336e0
Token supply	100,000,000,000
Token ticker	MR
Decimals	18
Token holders	1
Transaction count	1
Reward	0xe9e7cea3dedca5984780bafc599bd69add087d56
Dev wallet	0x26f3f79e5777c72de8432e438adcfb1c799064c9
Dividend tracker	0x17afdfeca49aedef3c8236944a21fcb0f3a3ea1cc
Marketing wallet	0x0a5e73df3836677eb7e22cb782e1cbfdc56da22a
Contract deployer address	0x49273B37ad4BbB7b85C292A540F39E4CAc9e6277
Contract's current owner address	0x49273b37ad4bbb7b85c292a540f39e4cac9e6277

Token distribution

Tokens are distributed as follows:










































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	
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	Selfdestruct function security	pass	
		Access control of owners	pass	
		Business logics	pass	
3	Business security	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












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 its 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 
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 		
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 		
IUniswapV2Factory	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 
IUniswapV2Router01	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¶
IUniswapV2Router02	Interface	IUniswapV2Router01		
L	removeLiquidityETHSupportingFeeOnTransferTokens	External ¶		NO¶
L	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ¶		NO¶
L	swapExactTokensForTokensSupport	External ¶		NO¶

	ingFeeOnTransfer Tokens			
L	swapExactETHFor TokensSupporting FeeOnTransferTo kens	External ¶		NO ¶
L	swapExactTokens ForETHSupporting FeeOnTransferTo kens	External ¶		NO ¶
IPinkAntiBot	Interface			
L	setTokenOwner	External ¶		NO ¶
L	onPreTransferChe ck	External ¶		NO ¶
IDividendDistributor	Interface			
L	setDistributionCrit eria	External ¶		NO ¶
L	setShare	External ¶		NO ¶
L	deposit	External ¶		NO ¶
L	process	External ¶		NO ¶
L	purge	External ¶		NO ¶
DividendDistributor	Implementation	IDividendDistributor		
L		Public ¶		NO ¶
L		External ¶		NO ¶



L	setDistributionCriteria	External 		onlyToken
L	purge	External 		onlyToken
L	setShare	External 		onlyToken
L	deposit	External 		onlyToken
L	process	External 		onlyToken
L	shouldDistribute	Internal 		
L	distributeDividend	Internal 		
L	claimDividend	External 		NO 
L	getUnpaidEarnings	Public 		NO 
L	getHolderDetails	Public 		NO 
L	getCumulativeDividends	Internal 		
L	getLastProcessedIndex	External 		NO 
L	getNumberOfTokenHolders	External 		NO 
L	getShareholdersList	External 		NO 
L	totalDistributedRewards	External 		NO 
L	addShareholder	Internal 		
L	removeShareholder	Internal 		

MRTOKEN	Implementation	IERC20, Ownable		
L		Public 		NO 
L		External 		NO 
L	totalSupply	External 		NO 
L	name	Public 		NO 
L	symbol	Public 		NO 
L	decimals	Public 		NO 
L	balanceOf	Public 		NO 
L	getHolderDetails	Public 		NO 
L	getLastProcessedIndex	Public 		NO 
L	getNumberOfTokenHolders	Public 		NO 
L	totalDistributedRewards	Public 		NO 
L	allowance	External 		NO 
L	approve	Public 		NO 
L	_approve	Internal 		
L	approveMax	External 		NO 
L	transfer	External 		NO 
L	transferFrom	External 		NO 

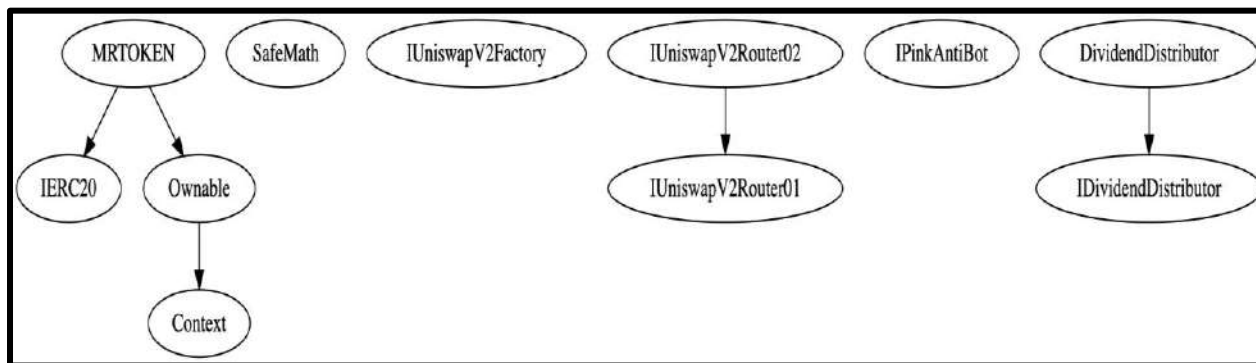
L	_transferFrom	Internal 		
L	_basicTransfer	Internal 		
L	shouldTakeFee	Internal 		
L	takeFee	Internal 		
L	shouldSwapBack	Internal 		
L	clearStuckBalance	External 		onlyOwner
L	changeSwapToken	External 		onlyOwner
L	updateBuyFees	Public 		onlyOwner
L	updateSellFees	Public 		onlyOwner
L	tradingStatus	Public 		onlyOwner
L	whitelistPreSale	Public 		onlyOwner
L	purgeBeforeSwitch	Public 		onlyOwner
L	includeMeinRewards	Public 		NO 
L	switchToken	Public 		onlyOwner
L	___claimRewards	Public 		NO 
L	claimProcess	Public 		NO 
L	swapBackInBnb	Internal 		swapping
L	swapBackInTokens	Internal 		swapping

L	swapAndSendFees	Private 		
L	swapAndLiquify	Private 		
L	swapTokensForEth	Private 		
L	swapTokensForTokens	Private 		
L	addLiquidity	Private 		
L	setIsDividendExempt	External 		onlyOwner
L	setIsFeeExempt	External 		onlyOwner
L	setFeeReceivers	External 		onlyOwner
L	setSwapBackSettings	External 		onlyOwner
L	setDistributionCriteria	External 		onlyOwner
L	setDistributorSettings	External 		onlyOwner
L	claimTokens	External 		onlyOwner
L	enableAntibot	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**
No low severity issues found.

Owner privileges

- ❖ The owner can get stuck BNB to the owner wallet from the contract.

```
ftrace | funcSig
function clearStuckBalance(uint256 amountPercentage↑) external onlyOwner {
    uint256 amountBNB = address(this).balance;
    payable(msg.sender).transfer((amountBNB * amountPercentage↑) / 100);
}
```

- ❖ The owner can change swap token (dev and marketing fee token).

```
ftrace | funcSig
function changeSwapToken(address token↑) external onlyOwner {
    SWAPTOKEN = token↑;
}
```

- ❖ The owner can change all buy and sell fees maximum up to 30%.

```
ftrace | funcSig
function updateBuyFees(
    uint256 reward↑,
    uint256 marketing↑,
    uint256 liquidity↑,
    uint256 dev↑,
    uint256 burn↑
) public onlyOwner {
    buyDividendRewardsFee = reward↑;
    buyMarketingFee = marketing↑;
    buyLiquidityFee = liquidity↑;
    buyDevFee = dev↑;
    buyBurnFee = burn↑;
    buyTotalFees = reward↑.add(marketing↑).add(liquidity↑).add(dev↑).add(burn↑);
    require(buyTotalFees <= 30, "Total Fee must be less than 30%");
}

ftrace | funcSig
function updateSellFees(
    uint256 reward↑,
    uint256 marketing↑,
    uint256 liquidity↑,
    uint256 dev↑,
    uint256 burn↑
) public onlyOwner {
    sellDividendRewardsFee = reward↑;
    sellMarketingFee = marketing↑;
    sellLiquidityFee = liquidity↑;
    sellDevFee = dev↑;
    sellBurnFee = burn↑;
    sellTotalFees = reward↑.add(marketing↑).add(liquidity↑).add(dev↑).add(burn↑);
    require(sellTotalFees <= 30, "Total Fee must be less than 30%");
}
```

- ❖ The owner can enable/disable trading.

```
// switch Trading
ftrace | funcSig
function tradingStatus(bool _status↑) public onlyOwner {
    tradingOpen = _status↑;
}
```

- ❖ The owner can whitelist pre-sale address.

```
ftrace | funcSig
function whitelistPreSale(address _preSale↑) public onlyOwner {
    isFeeExempt[_preSale↑] = true;
    isDividendExempt[_preSale↑] = true;
    isAuthorized[_preSale↑] = true;
}
```

- ❖ The owner can transfer dividend tracker tokens amount to owner wallet before changing the dividend token.

```
// new dividend tracker, clear balance
ftrace | funcSig
function purgeBeforeSwitch() public onlyOwner {
    dividendDistributor.purge(msg.sender);
}
```


- ❖ The owner can change the reward token.

```
// new dividend tracker
ftrace | funcSig
function switchToken(address rewardToken↑, bool isIncludeHolders↑)
    public
    onlyOwner
{
    require(rewardToken↑ != WBNE, "Can not reward BNB in this tracker");
    REWARD = rewardToken↑;
    // get current shareholders list
    address[] memory currentHolders = dividendDistributor
        .getShareHoldersList();
    dividendDistributor = new DividendDistributor(
        address(router),
        rewardToken↑
    );
    if (isIncludeHolders↑) {
        // add old share holders to new tracker
        for (uint256 i = 0; i < currentHolders.length; i++) {
            try
                dividendDistributor.setShare(
                    currentHolders[i],
                    balances[currentHolders[i]]
                )
            catch {}
        }
    }

    emit ChangeRewardTracker(rewardToken↑);
}
}
```

- ❖ The owner can include/exclude wallets from dividends.

```
ftrace | funcSig
function setIsDividendExempt(address holder↑, bool exempt↑)
    external
    onlyOwner
{
    require(holder↑ != address(this) && holder↑ != pair);
    isDividendExempt[holder↑] = exempt↑;
    if (exempt↑) {
        dividendDistributor.setShare(holder↑, 0);
    } else {
        dividendDistributor.setShare(holder↑, balances[holder↑]);
    }
}
}
```

- ❖ The owner can include/exclude wallets from fees.

```
ftrace | funcSig
function setIsFeeExempt(address holder↑, bool exempt↑) external onlyOwner {
    isFeeExempt[holder↑] = exempt↑;
}
```

- ❖ The owner can change dev and marketing wallets.

```
ftrace | funcSig
function setFeeReceivers(
    address _marketingFeeReceiver↑,
    address _devFeeReceiver↑
) external onlyOwner {
    marketingFeeReceiver = _marketingFeeReceiver↑;
    devFeeReceiver = _devFeeReceiver↑;
}
```

- ❖ The owner can enable/disable swapping and can change the swap point.

```
ftrace | funcSig
function setSwapBackSettings(bool _enabled↑, uint256 _amount↑)
    external
    onlyOwner
{
    swapEnabled = _enabled↑;
    swapThreshold = _amount↑;
}
```

- ❖ The owner can change minimum distribution time and minimum reward token amount.

```
ftrace | funcSig
function setDistributionCriteria(
    uint256 _minPeriod↑,
    uint256 _minDistribution↑
) external onlyOwner {
    dividendDistributor.setDistributionCriteria(
        _minPeriod↑,
        _minDistribution↑
    );
}
```

Functions publicly available for all users:

- ❖ Holders can get their last claim time, unpaid earnings, total rewards and their index in the tracker by passing wallets address.

```
ftrace | funcSig
function getHolderDetails(address holder↑)
    public
    view
    returns (
        uint256,
        uint256,
        uint256,
        uint256
    )
{
    return dividendDistributor.getHolderDetails(holder↑);
}
```

- ❖ Holders can enter to the new tracker by calling this function.

```
ftrace | funcSig
function includeMainRewards() public {
    require(
        !isDividendExempt[msg.sender],
        "You are not allowed to get rewards"
    );
    try
        dividendDistributor.setShare(msg.sender, balances[msg.sender])
    {} catch {}

    emit IncludeInReward(msg.sender);
}
```

- ❖ Holders can manually claim rewards.

```
ftrace | funcSig
function __claimRewards(bool tryAll↑) public {
    dividendDistributor.claimDividend();
    if (tryAll↑) {
        try dividendDistributor.process(distributorGas) {} catch {}
    }
}
```

Audit conclusion

While conducting the audit of the Meta Ruffy smart contract, it was observed that there is nothing alarming with the code.