Audit Report

Metafintec

February 2022

Type	BEP20
Network	BSC
Address	0x44921ab9fdf624433c75e7f75bb274e7095ddbfb
Audited by	© coinscope

Table of Contents

Contract Review

Contract Name	METAFINTEC
Compiler Version	v0.8.11+commit.d7f03943
Optimization	200 runs
Licence	MIT
Explorer	https://bscscan.com/token/0x44921ab9fdf624433c75e7f75bb274e7095ddbfb
Symbol	MAFTEC
Decimals	18
Total Supply	99,999,999,999
Source	contract.sol
Domain	

Audit Updates

Initial Audit	3rd March 2022
Corrected	

Contract Analysis

● Critical ● Medium ● Minor ● Pass

Severity	Code	Description
●	ST	Contract Owner is not able to stop or pause transactions
●	OCTD	Contract Owner is not able to transfer tokens from specific address
●	OTUT	Owner Transfer User's Tokens
●	ELFM	Contract Owner is not able to increase fees more than a reasonable percent (25%)
●	ULTW	Contract Owner is not able to increase the amount of liquidity taken by dev wallet more than a reasonable percent
●	MT	Contract Owner is not able to mint new tokens
●	BT	Contract Owner is not able to burn tokens from specific wallet
●	BC	Contract Owner is not able to blacklist wallets from selling

ST - Stop Transactions

Criticality	critical
Location	contract.sol#L1151,1195

Description

The contract owner can convert the contract into a honeypot behaviour by setting buybackFactor to a very high percent.

```
{
            if (balance > buyBackUpperLimit) { balance = buyBackUpperLimit; }
            buyBackTokens(balance.div(100).mul(buybackFactor));
        }
```

Recommendation

The contract could embody a check for not allowing setting `buybackFactor` less than reasonable amount. A suggested implementation could check that the maximum amount should be more than a fixed percentage of the total supply.

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

Contract Diagnostics

Critical Medium Minor

Severity	Code	Description
	L01	Public Function could be Declared External
	L02	State Variables could be Declared Constant
	L05	Unused State Variable
	L04	Conformance to Solidity Naming Conventions
	L09	Dead Code Elimination
	L12	Using Variables before Declaration
	L07	Missing Events Arithmetic
	L15	Local Scope Variable Shadowing
	L14	Uninitialized Variables in Local Scope
	L13	Divide before Multiply Operation

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L61,66,192,200,204,223,228,232,237,248 and 23 more

Description

Public functions that are never called by the contract should be declared external to save gas.

```
process
getAccountAtIndex
buyBackUpperLimitAmount
setMaxWalletLimit
setExcludedFromWhale
setSwapAndLiquifyEnabled
dividendTokenBalanceOf
withdrawableDividendOf
isExcludedFromFees
...
```

Recommendation

Use the external attribute for functions never called from the contract

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L743

Description

Constant state variables should be declared constant to save gas.

```
totalFees
```

Recommendation

Add the constant attribute to state variables that never change.

L05 - Unused State Variable

Criticality	minor
Location	contract.sol#L131

Description

There are segments that contain unused state variables.

```
MAX_INT256
```

Recommendation

Remove unused state variables.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L394,399,403,407,344,345,513,670,671,678 and 21 more

Description

Solidity defines a naming convention that should be followed. Rule exceptions:

- Allow constant variable name/symbol/decimals to be lowercase.
- Allow _ at the beginning of the mixed_case match for private variables and unused parameters.

```
_account
deadAddress
_marketingWalletAddress
buybackFee
marketingFee
liquidityFee
USDTRewardsFee
_maxTxAmount
USDT
...
```

Recommendation

Follow the Solidity naming convention.
https://docs.soliditylang.org/en/v0.4.25/style-guide.html#naming-conventions

L09 - Dead Code Elimination

Criticality	minor
Location	contract.sol#L37,412,118,123,156,139,132

Description

Functions that are not used in the contract, and make the code's size bigger.

```
mul
div
abs
mod
_transfer
_msgData
```

Recommendation

Remove unused functions.

L12 - Using Variables before Declaration

Criticality	minor
Location	contract.sol#L1008

Description

The contract is using a variable before the declaration. This is usually happening either if it has not been declared yet or the variable has been declared in a different scope.

```
iterations
claims
lastProcessedIndex
```

Recommendation

The variables should be declared before any usage of them.

L07 - Missing Events Arithmetic

Criticality	minor
Location	contract.sol#L874,1116,1167,1217,1221,1225

Description

Detected missing events for critical arithmetic parameters. There are functions that have no event emitted, so it is difficult to track off-chain changes.

```
buybackFactor = _percent
minBalanceForBuyback = _balanceInWei
buyBackUpperLimit = buyBackLimit * 10 ** 18
maxLimit = amount
_maxTxAmount = _amount
swapTokensAtAmount = _value
```

Recommendation

Emit an event for critical parameter changes.

L15 - Local Scope Variable Shadowing

Criticality	minor
Location	contract.sol#L352,394,399,403,407

Description

The are variables that are defined in the local scope containing the same name from an upper scope.

```
_owner
_symbol
_name
```

Recommendation

The local variables should have different names from the upper scoped variables.

L14 - Uninitialized Variables in Local Scope

Criticality	minor
Location	contract.sol#L1008

Description

The are variables that are defined in the local scope and are not initialized.

```
lastProcessedIndex
claims
iterations
```

Recommendation

All the local scoped variables should be initialized.

L13 - Divide before Multiply Operation

Criticality	minor
Location	contract.sol#L1033,1193,720

Description

Performing divisions before multiplications may cause lose of prediction.

```
maxLimit = _totalSupply.div(1000).mul(1)
buyBackTokens(balance.div(100).mul(buybackFactor))
bnbForLiquidity = newBalance.div(swapableFee).mul(liquidityFee).div(2)
halfLiquidityTokens = tokens.div(swapableFee).mul(liquidityFee).div(2)
```

Recommendation

The multiplications should be prior to the divisions.

Contract Functions

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
IBEP20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
IBEP20Metadata	Interface	IBEP20		
	name	External		-
	symbol	External		-
	decimals	External		-
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
SafeMath	Library			
	add	Internal		
	sub	Internal		

	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
	mod	Internal		
	mod	Internal		
SafeMathInt	Library			
	mul	Internal		
	div	Internal		
	sub	Internal		
	add	Internal		
	abs	Internal		
	toUint256Safe	Internal		
SafeMathUint	Library			
	toInt256Safe	Internal		
BEP20	Implementation	Context, IBEP20, IBEP20Metadata		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_mint	Internal	✓	

	_burn	Internal	✓	
	_approve	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
DividendPayingTokenInterface	Interface			
	dividendOf	External		-
	withdrawDividend	External	✓	-
DividendPayingTokenOptionalInterface	Interface			
	withdrawableDividendOf	External		-
	withdrawnDividendOf	External		-
	accumulativeDividendOf	External		-
DividendPayingToken	Implementation	BEP20, Ownable, DividendPayingTokenInterface, DividendPayingTokenOptionalInterface		
	<Constructor>	Public	✓	BEP20
	distributeUSDTDividends	Public	✓	onlyOwner
	withdrawDividend	Public	✓	-
	_withdrawDividendOfUser	Internal	✓	
	dividendOf	Public		-
	withdrawableDividendOf	Public		-
	withdrawnDividendOf	Public		-
	accumulativeDividendOf	Public		-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_setBalance	Internal	✓	

IterableMapping	Library			
	get	Public		-
	getIndexOfKey	Public		-
	getKeyAtIndex	Public		-
	size	Public		-
	set	Public	✓	-
	remove	Public	✓	-
IUniswapV2Router01	Interface			
	factory	External		-
	WETH	External		-
	addLiquidity	External	✓	-
	addLiquidityETH	External	Payable	-
	removeLiquidity	External	✓	-
	removeLiquidityETH	External	✓	-
	removeLiquidityWithPermit	External	✓	-
	removeLiquidityETHWithPermit	External	✓	-
	swapExactTokensForTokens	External	✓	-
	swapTokensForExactTokens	External	✓	-
	swapExactETHForTokens	External	Payable	-
	swapTokensForExactETH	External	✓	-
	swapExactTokensForETH	External	✓	-
	swapETHForExactTokens	External	Payable	-
	quote	External		-
	getAmountOut	External		-
	getAmountIn	External		-
	getAmountsOut	External		-
	getAmountsIn	External		-
IUniswapV2Router02	Interface	IUniswapV2Router01		
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-

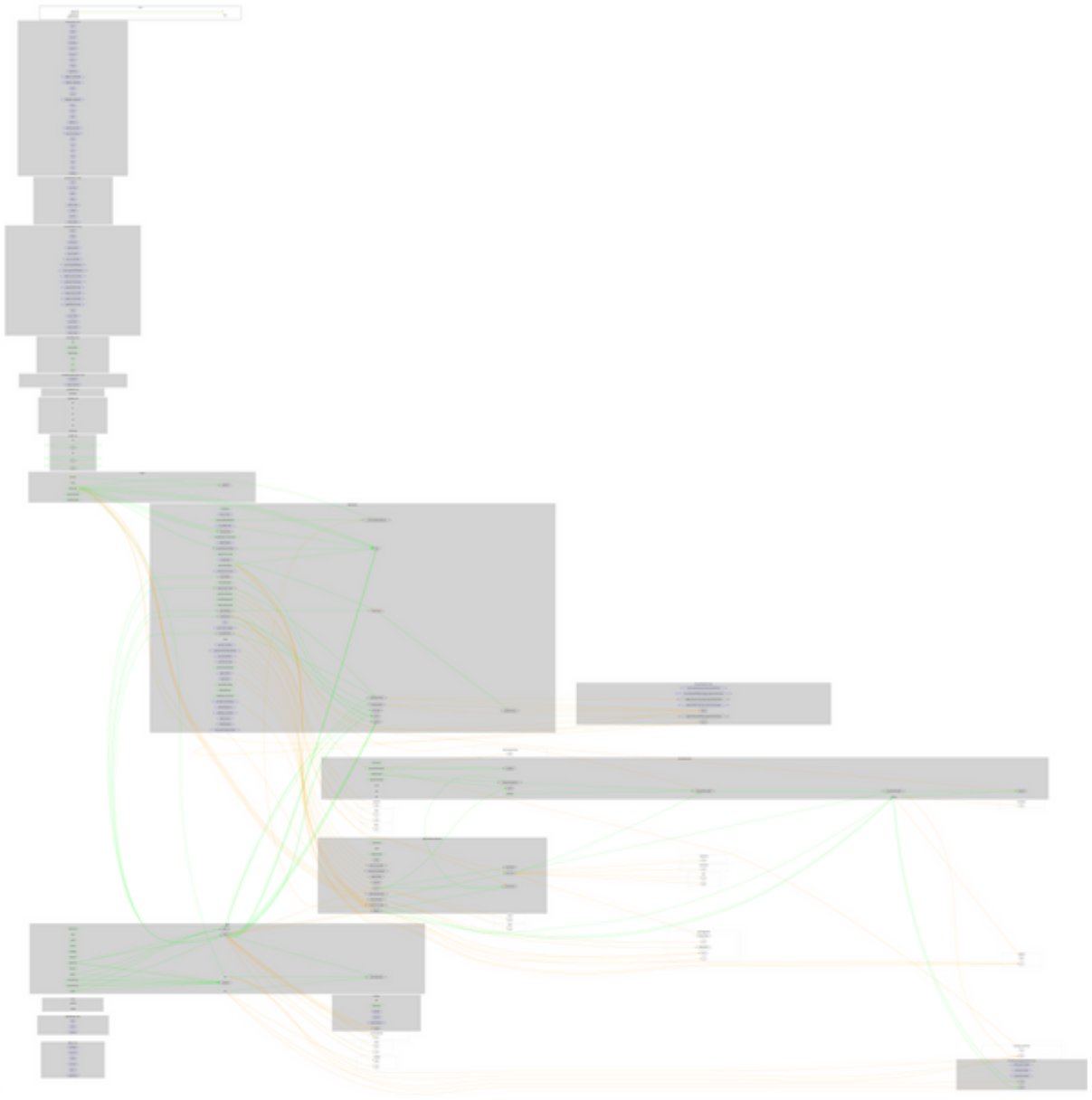
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
IUniswapV2Factory	Interface			
	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-
IUniswapV2Pair	Interface			
	name	External		-
	symbol	External		-
	decimals	External		-
	totalSupply	External		-
	balanceOf	External		-
	allowance	External		-
	approve	External	✓	-
	transfer	External	✓	-
	transferFrom	External	✓	-
	DOMAIN_SEPARATOR	External		-
	PERMIT_TYPEHASH	External		-
	nonces	External		-
	permit	External	✓	-
	MINIMUM_LIQUIDITY	External		-
	factory	External		-
	token0	External		-
	token1	External		-

	getReserves	External		-
	price0CumulativeLast	External		-
	price1CumulativeLast	External		-
	kLast	External		-
	mint	External	✓	-
	burn	External	✓	-
	swap	External	✓	-
	skim	External	✓	-
	sync	External	✓	-
	initialize	External	✓	-
LockToken	Implementation	Ownable		
	<Constructor>	Public	✓	-
	openTrade	External	✓	onlyOwner
	includeToWhiteList	External	✓	onlyOwner
METAFINTEC	Implementation	BEP20, LockToken		
	<Constructor>	Public	✓	BEP20
	<Receive Ether>	External	Payable	-
	updateDividendTracker	Public	✓	onlyOwner
	updateUniswapV2Router	Public	✓	onlyOwner
	excludeFromFees	Public	✓	onlyOwner
	excludeMultipleAccountsFromFees	Public	✓	onlyOwner
	setMarketingWallet	External	✓	onlyOwner
	setAutomatedMarketMakerPair	Public	✓	onlyOwner
	_setAutomatedMarketMakerPair	Private	✓	
	updateGasForProcessing	Public	✓	onlyOwner
	updateClaimWait	External	✓	onlyOwner
	getClaimWait	External		-
	setSwapTokensAtAmount	External	✓	onlyOwner
	getTotalDividendsDistributed	External		-
	isExcludedFromFees	Public		-
	withdrawableDividendOf	Public		-
	dividendTokenBalanceOf	Public		-
	excludeFromDividends	External	✓	onlyOwner

	getAccountDividendsInfo	External		-
	getAccountDividendsInfoAtIndex	External		-
	processDividendTracker	External	✓	-
	claim	External	✓	-
	getLastProcessedIndex	External		-
	getNumberOfDividendTokenHolders	External		-
	_transfer	Internal	✓	open
	swapAndSendToFee	Private	✓	
	recoverBEP20Tokens	External	✓	onlyOwner
	swapAndLiquify	Private	✓	
	swapTokensForEth	Private	✓	
	swapTokensForUSDT	Private	✓	
	addLiquidity	Private	✓	
	swapAndSendDividends	Private	✓	
	setMaxTxAmount	External	✓	onlyOwner
	setSwapAndLiquifyEnabled	Public	✓	onlyOwner
	excludeWalletsFromWhales	Private	✓	
	checkForWhale	Private		
	setExcludedFromWhale	Public	✓	onlyOwner
	setMaxWalletLimit	Public	✓	onlyOwner
	buyBackUpperLimitAmount	Public		-
	buyBackTokens	Private	✓	
	checkForBuyBack	Private	✓	
	swapETHForTokens	Private	✓	
	setBuybackUpperLimit	External	✓	onlyOwner
	setMinBalanceForBuyback	External	✓	onlyOwner
	setBuybackFactor	External	✓	onlyOwner
	setBuyBackEnabled	External	✓	onlyOwner
	manualBuyback	External	✓	onlyOwner
METAFINTECDividendTracker	Implementation	Ownable, DividendPayingToken		
	<Constructor>	Public	✓	DividendPayingToken
	_transfer	Internal		

	withdrawDividend	Public		-
	excludeFromDividends	External	✓	onlyOwner
	updateClaimWait	External	✓	onlyOwner
	getLastProcessedIndex	External		-
	getNumberOfTokenHolders	External		-
	getAccount	Public		-
	getAccountAtIndex	Public		-
	canAutoClaim	Private		
	setBalance	External	✓	onlyOwner
	process	Public	✓	-
	processAccount	Public	✓	onlyOwner

Contract Flow



Domain Info

Domain Name	metafintec.org
Registry Domain ID	D402200000018719582-LROR
Creation Date	2022-01-01T18:13:48Z
Updated Date	2022-01-09T17:56:24Z
Registry Expiry Date	2023-01-01T18:13:48Z
Registrar WHOIS Server	whois.ionos.com
Registrar URL	https://registrar.ionos.info/
Registrar	1&1 IONOS SE
Registrar IANA ID	83

The domain has been created about 2 months before the creation of the audit. It will expire in 10 months.

There is no public billing information, the creator is protected by the privacy settings.

Summary

The Smart Contract analysis reported no compiler error and only one critical threat issue. The contract can be converted into a honeypot and prevent users from selling by abusing the admin functions. The transaction fees are fixed to 14% and the maximum transaction amount is fixed to 1% of the total supply. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats.

Disclaimer

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment.

Coinscope team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed.

The Coinscope team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Coinscope receive a payment to manipulate those results or change the awarding badge that we will be adding in our website.

Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token.

The Coinscope team disclaims any liability for the resulting losses.

About Coinscope

Coinscope is the leading early coin listing, voting and auditing authority firm. The audit process is analyzing and monitoring many aspects of the project. That way, it gives the community a good sense of security using an informative report and a generic score.

Coinscope is aiming to make crypto discoverable and efficient globally. It provides all the essential tools to assist users draw their own conclusions.

The Coinscope.co team

https://www.coinscope.co