Audit Report

Moria Universe

January 2022

Type	BEP20
Network	BSC
Address	0xD05DDfA01427B6c7eA7CD988a10808244566B993
Audited by	© coinscope

Table of Contents

Contract Review

Contract Name	MoriaUniverse
Compiler Version	v0.8.4+commit.c7e474f2
Optimization	200 runs
Licence	None
Explorer	https://bscscan.com/token/0xD05DDfA01427B6c7eA7CD988a10808244566B993
Symbol	MORIA
Decimals	9
Total Supply	10,000,000
Source	contract.sol
Domain	

Audit Updates

Initial Audit	27th January 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#L1094

Description

The contract owner has the authority to stop transactions for all users excluding the owner. The owner may take advantage of it by setting the `_maxTxAmount` to zero.

```
       if(from != owner() && to != owner()) {
            require(amount <= _maxTxAmount, "Transfer amount exceeds the
maxTxAmount.");
        }
```

Recommendation

The contract could embody a check for not allowing setting the _maxTxAmount less than a 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.

ELFM - Exceed Limit Fees Manipulation

Criticality	critical
Location	contract.sol#L1076

Description

The contract owner has the authority to increase total fees over the allowed limit of 25%. The owner may take advantage of it by calling the `setSellFee` function with a high percentage value. This will have as a result the contract to behave like a honeypot.

```
    function setBuyFee(uint256 buyTaxFee, uint256 buyLiquidityFee) external
onlyOwner {
        _buyTaxFee = buyTaxFee;
        _buyLiquidityFee = buyLiquidityFee;
    }

    function setSellFee(uint256 sellTaxFee, uint256 sellLiquidityFee) external
onlyOwner {
        _sellTaxFee = sellTaxFee;
        _sellLiquidityFee = sellLiquidityFee;
    }
```

Recommendation

The contract could embody a check for the maximum acceptable value.

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
	L04	Conformance to Solidity Naming Conventions
	L09	Dead Code Elimination
	L07	Missing Events Arithmetic

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L1158,L1139,L1116 and 29 more

Description

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

```
transferForeignToken
changeRouterVersion
setAutoBuyBackEnabled
...
```

Recommendation

Use the external attribute for functions never called from the contract.

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L426,L431,L430 and 2 more

Description

Constant state variables should be declared constant to save gas.

```
_tTotal
_symbol
_name
...
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L489,L480,L479 and 48 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.

```
_isEnabledBuyBackAndBurn
_buyBackMaxTimeForHistories
_buyBackTimeInterval
...
```

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#L79,L75,L840 and 8 more

Description

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

```
mod
addLiquidity
_msgData
...
```

Recommendation

Remove unused functions.

L07 - Missing Events Arithmetic

Criticality	minor
Location	contract.sol#L1098,L1094,L1090 and 10 more

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.

```
minimumTokensBeforeSwap = _minimumTokensBeforeSwap
marketingDivisor = divisor
_maxTxAmount = maxTxAmount
...
```

Recommendation

Emit an event for critical parameter changes.

Contract Functions

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
SafeMath	Library			
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
	mod	Internal		
	mod	Internal		
Address	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	

	_functionCallWithValue	Private	✓	
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	getUnlockTime	Public		-
	getTime	Public		-
	lock	Public	✓	onlyOwner
	unlock	Public	✓	-
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		-

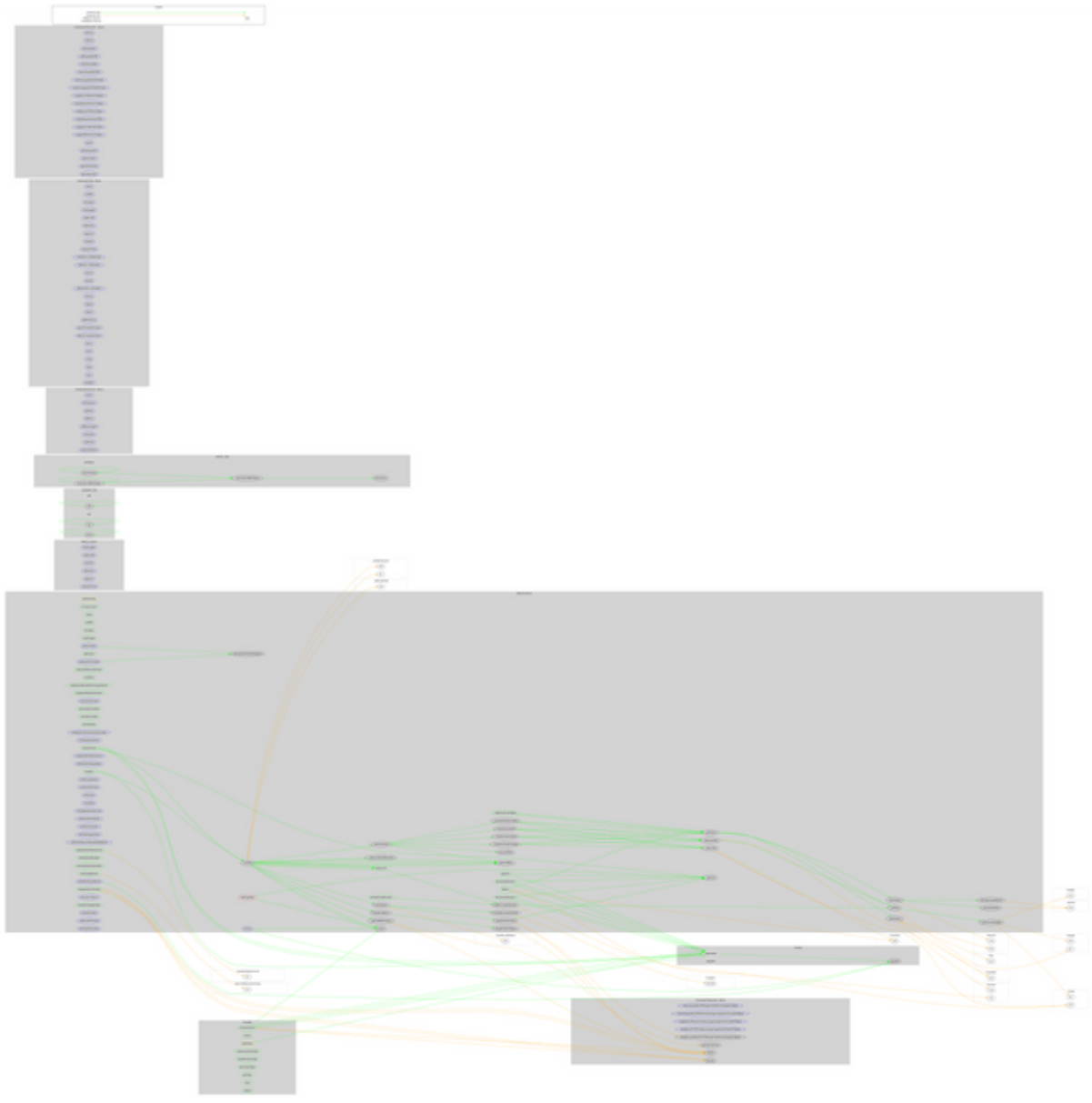
	nonces	External		-
	permit	External	✓	-
	MINIMUM_LIQUIDITY	External		-
	factory	External		-
	token0	External		-
	token1	External		-
	getReserves	External		-
	price0CumulativeLast	External		-
	price1CumulativeLast	External		-
	kLast	External		-
	burn	External	✓	-
	swap	External	✓	-
	skim	External	✓	-
	sync	External	✓	-
	initialize	External	✓	-
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	✓	-
MoriaUniverse	Implementation	Context, IERC20, Ownable		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	isExcludedFromReward	Public		-
	totalFees	Public		-
	minimumTokensBeforeSwapAmount	Public		-
	buyBackSellLimitAmount	Public		-
	deliver	Public	✓	-
	reflectionFromToken	Public		-
	tokenFromReflection	Public		-
	excludeFromReward	Public	✓	onlyOwner

	includeInReward	External	✓	onlyOwner
	_approve	Private	✓	
	_transfer	Private	✓	
	swapTokens	Private	✓	lockTheSwap
	buyBackTokens	Private	✓	lockTheSwap
	swapTokensForEth	Private	✓	
	swapETHForTokens	Private	✓	
	addLiquidity	Private	✓	
	_tokenTransfer	Private	✓	
	_transferStandard	Private	✓	
	_transferToExcluded	Private	✓	
	_transferFromExcluded	Private	✓	
	_transferBothExcluded	Private	✓	
	_reflectFee	Private	✓	
	_getValues	Private		
	_getTValues	Private		
	_getRValues	Private		
	_getRate	Private		
	_getCurrentSupply	Private		
	_takeLiquidity	Private	✓	
	calculateTaxFee	Private		
	calculateLiquidityFee	Private		
	removeAllFee	Private	✓	
	restoreAllFee	Private	✓	
	isExcludedFromFee	Public		-
	excludeFromFee	Public	✓	onlyOwner
	includeInFee	Public	✓	onlyOwner
	_getSellBnBAmount	Private		
	_removeOldSellHistories	Private	✓	
	SetBuyBackMaxTimeForHistories	External	✓	onlyOwner
	SetBuyBackDivisor	External	✓	onlyOwner
	GetBuyBackTimeInterval	Public		-
	SetBuyBackTimeInterval	External	✓	onlyOwner
	SetBuyBackRangeRate	External	✓	onlyOwner
	GetSwapMinutes	Public		-

	SetSwapMinutes	External	✓	onlyOwner
	setTaxFeePercent	External	✓	onlyOwner
	setBuyFee	External	✓	onlyOwner
	setSellFee	External	✓	onlyOwner
	setLiquidityFeePercent	External	✓	onlyOwner
	setBuyBackSellLimit	External	✓	onlyOwner
	setMaxTxAmount	External	✓	onlyOwner
	setMarketingDivisor	External	✓	onlyOwner
	setNumTokensSellToAddToBuyBack	External	✓	onlyOwner
	setMarketingAddress	External	✓	onlyOwner
	setSwapAndLiquifyEnabled	Public	✓	onlyOwner
	setBuyBackEnabled	Public	✓	onlyOwner
	setAutoBuyBackEnabled	Public	✓	onlyOwner
	prepareForPreSale	External	✓	onlyOwner
	afterPreSale	External	✓	onlyOwner
	transferToAddressETH	Private	✓	
	changeRouterVersion	Public	✓	onlyOwner
	<Receive Ether>	External	Payable	-
	transferForeignToken	Public	✓	onlyOwner
	Sweep	External	✓	onlyOwner
	setAddressFee	External	✓	onlyOwner
	setBuyAddressFee	External	✓	onlyOwner
	setSellAddressFee	External	✓	onlyOwner

Contract Flow



Summary

The Smart Contract analysis reported no compiler errors and 2 critical threat issues. There are some functions that can be abused by the owner, like manipulating fees up to 100%, abusing the contract to behave like a honeypot and stopping transactions for everyone else except the owner. 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