COINSCOPE

Audit Report

PEPPA PIG MAFIA

December 2021

Type	BEP20
Address	-
File SHA256	2c7a92c06959b3a74e795624ea4edcf39e1f5d6084513e5ac1a0ae07ebb85c9b
Audited by	© coinscope

Table of Contents

Contract Review

The contract is not deployed in the BSC network. Thus, the audit uses the solidity source code with SHA256: `2c7a92c06959b3a74e795624ea4edcf39e1f5d6084513e5ac1a0ae07ebb85c9b`

Since it’s not deployed, there are some information that are missing from the report like the symbol, the contract name, the optimization algorithm etc.

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

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
●	L03	Redundant Statements
●	L04	Conformance to Solidity Naming Conventions
●	L09	Dead Code Elimination
●	L07	Missing Events Arithmetic
●	L08	Tautology or Contradiction

L01 - Public Function could be Declared External

Criticality	minor
Location	contract.sol#L474,L465,L186 and 8 more

Description

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

```
transferOwnership
renounceOwnership
decreaseAllowance
...
```

Recommendation

Use the external attribute for functions never called from the contract

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract.sol#L1151,L1134,L1136

Description

Constant state variables should be declared constant to save gas.

```
manualBurnFrequency
liquidityActiveBlock
earlyBuyPenaltyEnd
```

Recommendation

Add the constant attribute to state variables that never change.

L05 - Unused State Variable

Criticality	minor
Location	contract.sol#L485

Description

There are segments that contains unused state variable.

```
SafeMathInt
```

Recommendation

Remove unused state variables.

L03 - Redundant Statements

Criticality	minor
Location	contract.sol#L11

Description

Detect the usage of redundant statements that have no effect.

```
Context
```

Recommendation

Remove redundant statements if they congest code but offer no value.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract.sol#L1181,L1154,L1739 and 18 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.

```
_isExcludedMaxTransactionAmount
feeDivisor
_Enabled
...
```

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#L490,L502,L531 and 4 more

Description

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

```
mul
div
abs
...
```

Recommendation

Remove unused functions.

L07 - Missing Events Arithmetic

Criticality	minor
Location	contract.sol#L1739,L1348,L1339 and 2 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.

```
lpBurnFrequency = _frequencyInSeconds
marketingSellFee = _marketingFee
marketingBuyFee = _marketingFee
...
```

Recommendation

Emit an event for critical parameter changes.

L08 - Tautology or Contradiction

Criticality	minor
Location	contract.sol#L1739

Description

Detects expressions that are tautologies or contradictions. For instance, an uint variable will always be greater than or equal to zero.

```
require(bool,string)(_percent <= 1000 && _percent >= 0,Must set auto LP burn percent between
0% and 10%)
```

Recommendation

Fix the incorrect comparison by changing the value type or the comparison.

Contract Functions

Contract	**Type**	**Bases**		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
IUniswapV2Pair	Interface			
	sync	External	✓	-
IUniswapV2Factory	Interface			
	createPair	External	✓	-
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
IERC20Metadata	Interface	IERC20		
	name	External		-
	symbol	External		-
	decimals	External		-
ERC20	Implementation	Context, IERC20, IERC20Metadata		
	<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	✓	
	_createInitialSupply	Internal	✓	
	_approve	Internal	✓	
DividendPayingTokenOptionalInterface	Interface			
	withdrawableDividendOf	External		-
	withdrawnDividendOf	External		-
	accumulativeDividendOf	External		-
DividendPayingTokenInterface	Interface			
	dividendOf	External		-
	distributeDividends	External	Payable	-
	withdrawDividend	External	✓	-
SafeMath	Library			
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
	mod	Internal		

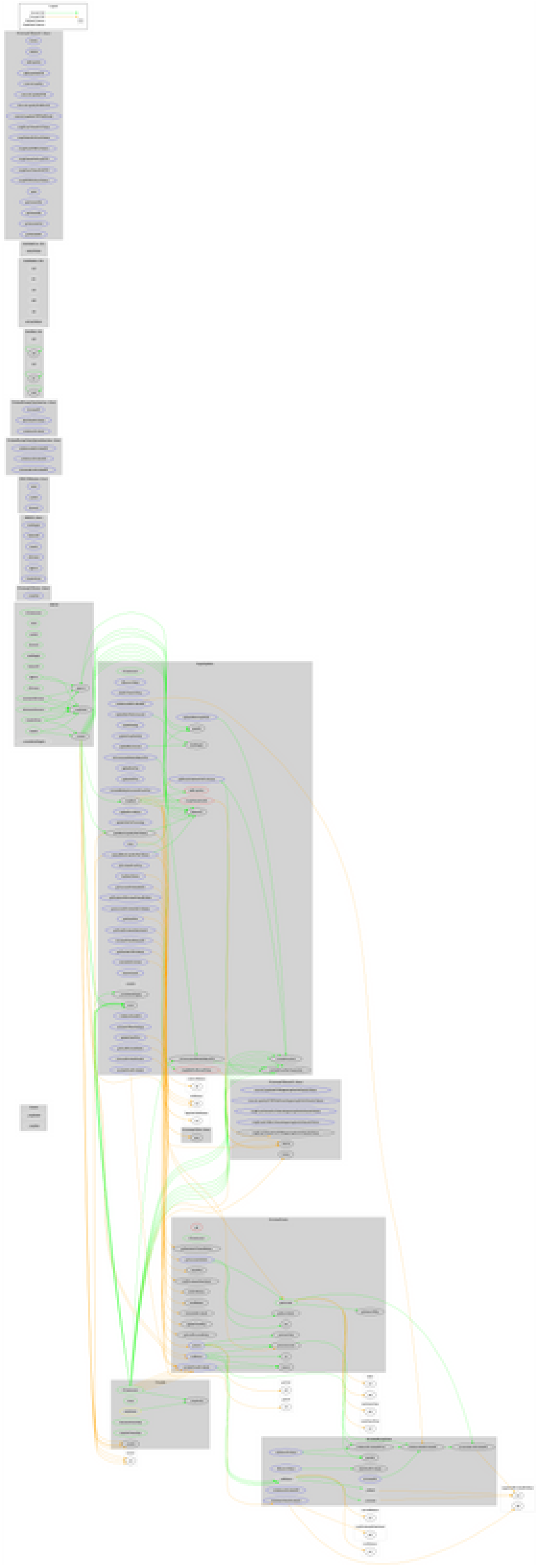
	mod	Internal		
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
SafeMathInt	Library			
	mul	Internal		
	div	Internal		
	sub	Internal		
	add	Internal		
	abs	Internal		
	toUint256Safe	Internal		
SafeMathUint	Library			
	toInt256Safe	Internal		
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	✓	-
DividendPayingToken	Implementation	DividendPayingTokenInterface, DividendPayingTokenOptionalInterface, Ownable		
	<Receive Ether>	External	Payable	-
	distributeDividends	Public	Payable	-
	distributeTokenDividends	External	✓	onlyOwner
	withdrawDividend	External	✓	-
	_withdrawDividendOfUser	Internal	✓	
	dividendOf	External		-
	withdrawableDividendOf	Public		-
	withdrawnDividendOf	External		-
	accumulativeDividendOf	Public		-
	_increase	Internal	✓	
	_reduce	Internal	✓	
	_setBalance	Internal	✓	
DividendTracker	Implementation	DividendPayingToken		

	get	Private		
	getIndexOfKey	Private		
	getKeyAtIndex	Private		
	size	Private		
	set	Private	✓	
	remove	Private	✓	
	<Constructor>	Public	✓	-
	excludeFromDividends	External	✓	onlyOwner
	includeInDividends	External	✓	onlyOwner
	updateClaimWait	External	✓	onlyOwner
	getLastProcessedIndex	External		-
	getNumberOfTokenHolders	External		-
	getAccount	Public		-
	getAccountAtIndex	External		-
	canAutoClaim	Private		
	setBalance	External	✓	onlyOwner
	process	External	✓	-
	processAccount	Public	✓	onlyOwner
PeppaPigMafia	Implementation	ERC20, Ownable		
	<Constructor>	Public	✓	ERC20
	<Receive Ether>	External	Payable	-
	addPresaleAddressForExclusions	External	✓	onlyOwner
	disableTransferDelay	External	✓	onlyOwner
	excludeFromDividends	External	✓	onlyOwner
	includeInDividends	External	✓	onlyOwner
	enableTrading	External	✓	onlyOwner
	updateSwapEnabled	External	✓	onlyOwner
	updateMaxAmount	External	✓	onlyOwner
	updateMaxWalletAmount	External	✓	onlyOwner
	updateBuyFees	External	✓	onlyOwner
	updateSellFees	External	✓	onlyOwner
	excludeFromMaxTransaction	Public	✓	onlyOwner
	excludeFromFees	Public	✓	onlyOwner
	excludeMultipleAccountsFromFees	External	✓	onlyOwner

	setAutomatedMarketMakerPair	External	✓	onlyOwner
	_setAutomatedMarketMakerPair	Private	✓	
	updateMarketingWallet	External	✓	onlyOwner
	updateDevAddress	External	✓	onlyOwner
	updateGasForProcessing	External	✓	onlyOwner
	updateClaimWait	External	✓	onlyOwner
	getClaimWait	External		-
	getTotalDividendsDistributed	External		-
	isExcludedFromFees	External		-
	withdrawableDividendOf	External		-
	dividendTokenBalanceOf	External		-
	getAccountDividendsInfo	External		-
	getAccountDividendsInfoAtIndex	External		-
	processDividendTracker	External	✓	-
	claim	External	✓	-
	getLastProcessedIndex	External		-
	getNumberOfDividendTokenHolders	External		-
	getNumberOfDividends	External		-
	removeLimits	External	✓	onlyOwner
	_transfer	Internal	✓	
	swapBnbForRewardToken	Private	✓	
	swapTokensForEth	Private	✓	
	addLiquidity	Private	✓	
	swapBack	Private	✓	
	buyBackTokens	External	✓	onlyOwner
	withdrawStuckEth	External	✓	onlyOwner
	setAutoLPBurnSettings	External	✓	onlyOwner
	autoBurnLiquidityPairTokens	Internal	✓	
	manualBurnLiquidityPairTokens	External	✓	onlyOwner

Contract Flow



Summary

The contract analysis did not report any compiler error or critical issue. The smart contract contains features like distribution and auto-burn. Users that behave like bots, will have a hard penalty in the trades opening. There are some informative comments that do not affect the contract security.

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