auditace.tech

AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

PEPACOIN

DATED : 3 May 23'

AUDIT SUMMARY

Project name – PEPACOIN

Date: 3 May, 2023

Scope of Audit- Audit Ace was consulted to conduct the smart contract audit of the solidity source codes.

Audit Status: Passed

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	0	0	0	0
Acknowledged	0	0	0	0	0
Resolved	0	0	0	0	0

USED TOOLS

Tools:

1- Manual Review:
A line by line code review has been performed by audit ace team.

2- BSC Test Network: All tests were conducted on the BSC Test network, and each test has a corresponding transaction attached to it. These tests can be found in the "Functional Tests" section of the report.

3- Slither :
The code has undergone static analysis using Slither.

Testnet version:
The tests were performed using the contract deployed on the BSC Testnet, which can be found at the following address:
https://testnet.bscscan.com/token/0x434ab6e14b1dee9bd075a94ec87701312131l8d0

Token Information

Token Name : PEPACOIN

Token Symbol: PEPA

Decimals: 18

Token Supply: 420,000,000,000,000

Token Address:
0x98E039F9673F69CeC1CcA175f3458AB3ff9a0751

Checksum:
7b0d6058387825c26af841b8913f24278a52984c

Owner:
0xdA2D4D9F2a7C7E9F1997FE56c6F05088fb9a6C6f
(at time of writing the audit)

Deployer:
0xdA2D4D9F2a7C7E9F1997FE56c6F05088fb9a6C6f

TOKEN OVERVIEW

Fees:

Buy Fees: 0%

Sell Fees: 0%

Transfer Fees: 0%

Fees Privilege: Owner

Ownership: Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: Yes

Blacklist: No

Other Privileges: updating fee - updating max tx and wallet

AUDIT METHODOLOGY

The auditing process will follow a routine as special considerations by Auditace:

- Review of the specifications, sources, and instructions provided to Auditace to make sure the contract logic meets the intentions of the client without exposing the user's funds to risk.
- Manual review of the entire codebase by our experts, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
- Specification comparison is the process of checking whether the code does what the specifications, sources, and instructions provided to Auditace describe.
- Test coverage analysis determines whether the test cases are covering the code and how much code isexercised when we run the test cases.
- Symbolic execution is analysing a program to determine what inputs cause each part of a program to execute.
- Reviewing the codebase to improve maintainability, security, and control based on the established industry and academic practices.

VULNERABILITY CHECKLIST

- Return values of low-level calls
- Private modifier
- Multiple Sends
- Using Suicide
- Gas Limitand Loops
- Address hardcoded
- Exception Disorder
- Using inline assembly
- Divide before multiply
- Missing Zero Address Validation
- Compiler version not fixed
- **Gasless Send**
- Using block.timestamp
- Re-entrancy
- Tautology or contradiction
- Timestamp Dependence
- Revert/require functions
- Use of tx.origin
- Integer overflow/underflow
- Dangerous strict equalities
- Using SHA3
- Using throw

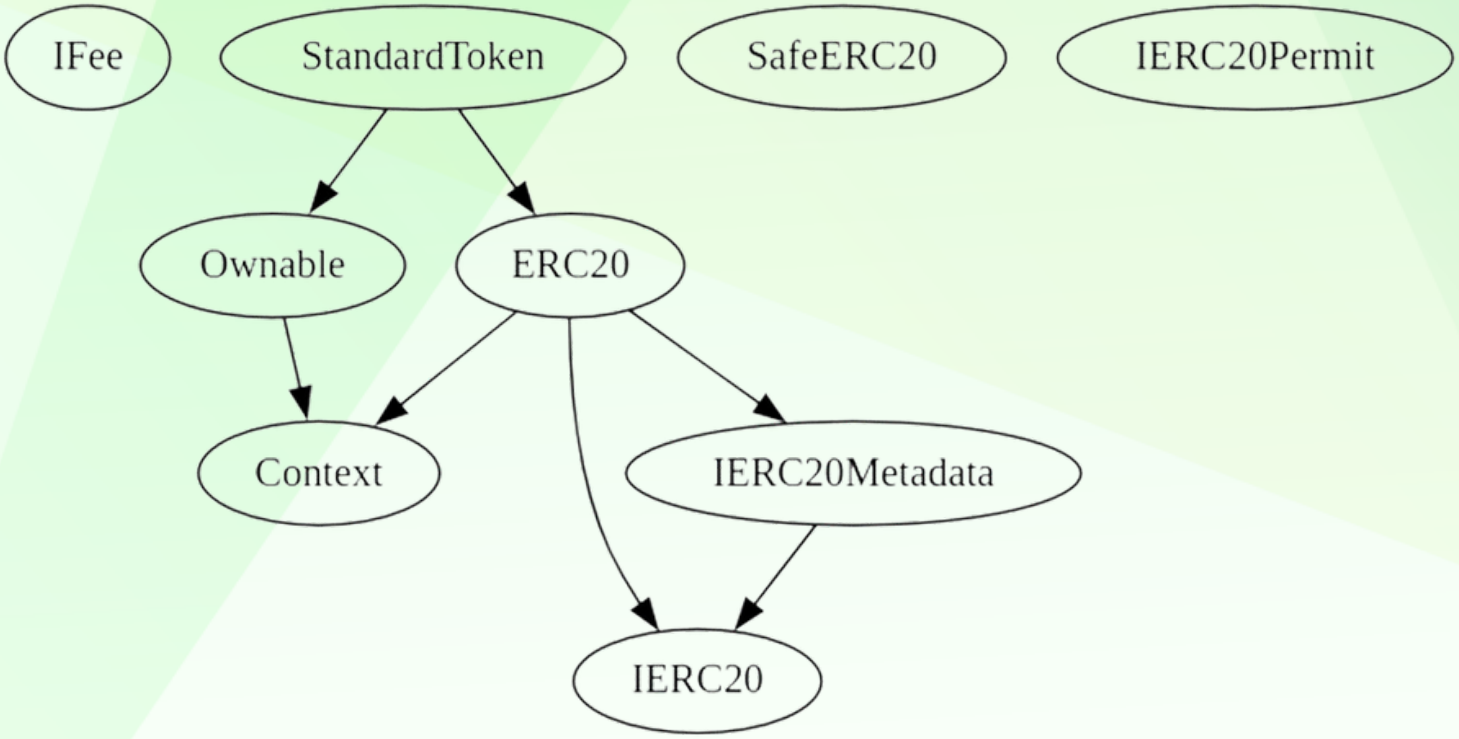
CLASSIFICATION OF RISK

Severity	Description
Critical	These vulnerabilities could be exploited easily and can lead to asset loss, data loss, asset, or data manipulation. They should be fixed right away.
High-Risk	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.
Medium-Risk	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.
Low-Risk	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.
Gas Optimization /Suggestion	A vulnerability that has an informational character but is not affecting any of the code.

Findings

Severity	Found
Critical	0
High-Risk	0
Medium-Risk	0
Low-Risk	0
Gas Optimization / Suggestions	0

INHERITANCE TREE



POINTS TO NOTE

- Owner is not able to set buy/sell/transfer tax (ownership renounced)
- Owner is not able to blacklist an arbitrary wallet
- Owner is not able to mint new tokens
- Owner is not able to disable buy/sell/transfer (ownership renounced)
- Owner is not able to set max wallet/tx amount (ownership renounced)

CONTRACT ASSESMENT

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
IUniswapV2Router01	Interface				
└	factory	External ❗		NO ❗
└	WETH	External ❗		NO ❗
└	addLiquidity	External ❗	🛑	NO ❗
└	addLiquidityETH	External ❗	💵	NO ❗
└	removeLiquidity	External ❗	🛑	NO ❗
└	removeLiquidityETH	External ❗	🛑	NO ❗
└	removeLiquidityWithPermit	External ❗	🛑	NO ❗
└	removeLiquidityETHWithPermit	External ❗	🛑	NO ❗
└	swapExactTokensForTokens	External ❗	🛑	NO ❗
└	swapTokensForExactTokens	External ❗	🛑	NO ❗
└	swapExactETHForTokens	External ❗	💵	NO ❗
└	swapTokensForExactETH	External ❗	🛑	NO ❗
└	swapExactTokensForETH	External ❗	🛑	NO ❗
└	swapETHForExactTokens	External ❗	💵	NO ❗
└	quote	External ❗		NO ❗
└	getAmountOut	External ❗		NO ❗
└	getAmountIn	External ❗		NO ❗
└	getAmountsOut	External ❗		NO ❗
└	getAmountsIn	External ❗		NO ❗
IUniswapV2Router02	Interface	IUniswapV2Router01			
└	removeLiquidityETHSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO ❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
IUniswapV2Pair	Interface				
└	name	External ❗		NO ❗
└	symbol	External ❗		NO ❗
└	decimals	External ❗		NO ❗
└	totalSupply	External ❗		NO ❗
└	balanceOf	External ❗		NO ❗
└	allowance	External ❗		NO ❗
└	approve	External ❗	🛑	NO ❗
└	transfer	External ❗	🛑	NO ❗
└	transferFrom	External ❗	🛑	NO ❗
└	DOMAIN_SEPARATOR	External ❗		NO ❗
└	PERMIT_TYPEHASH	External ❗		NO ❗

CONTRACT ASSESMENT

└	nonces	External ❗	|NO ❗
└	permit	External ❗	🛑 |NO ❗
└	MINIMUM_LIQUIDITY	External ❗	|NO ❗
└	factory	External ❗	|NO ❗
└	token0	External ❗	|NO ❗
└	token1	External ❗	|NO ❗
└	getReserves	External ❗	|NO ❗
└	price0CumulativeLast	External ❗	|NO ❗
└	price1CumulativeLast	External ❗	|NO ❗
└	kLast	External ❗	|NO ❗
└	mint	External ❗	🛑 |NO ❗
└	burn	External ❗	🛑 |NO ❗
└	swap	External ❗	🛑 |NO ❗
└	skim	External ❗	🛑 |NO ❗
└	sync	External ❗	🛑 |NO ❗
└	initialize	External ❗	🛑 |NO ❗
IUniswapV2Factory	Interface	|||		
└	feeTo	External ❗	|NO ❗
└	feeToSetter	External ❗	|NO ❗
└	getPair	External ❗	|NO ❗
└	allPairs	External ❗	|NO ❗
└	allPairsLength	External ❗	|NO ❗
└	createPair	External ❗	🛑 |NO ❗
└	setFeeTo	External ❗	🛑 |NO ❗
└	setFeeToSetter	External ❗	🛑 |NO ❗
└	INIT_CODE_PAIR_HASH	External ❗	|NO ❗
IUniswapV2Caller	Interface	|||		
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🛑 |NO ❗
IFee	Interface	|||		
└	payFee	External ❗	💵 |NO ❗
StandardToken	Implementation	ERC20, Ownable |||		
└	<Constructor>	Public ❗	💵	ERC20
└	decimals	Public ❗	|NO ❗
└	updateUniswapV2Pair	External ❗	🛑	onlyOwner
└	updateUniswapV2Router	Public ❗	🛑	onlyOwner
└	updateLiquidityFee	External ❗	🛑	onlyOwner
└	updateMaxWallet	External ❗	🛑	onlyOwner
└	updateMaxTransactionAmount	External ❗	🛑	onlyOwner
└	updateMarketingFee	External ❗	🛑	onlyOwner

CONTRACT ASSESMENT

└	updateMarketingWallet	External ❗	🛑	onlyOwner
└	updateMinAmountToTakeFee	External ❗	🛑	onlyOwner
└	setAutomatedMarketMakerPair	Public ❗	🛑	onlyOwner
└	_setAutomatedMarketMakerPair	Private 🔐	🛑	
└	excludeFromFee	External ❗	🛑	onlyOwner
└	excludeFromMaxTransactionAmount	External ❗	🛑	onlyOwner
└	_transfer	Internal 🔒	🛑	
└	takeFee	Private 🔐	🛑	lockTheSwap
└	swapTokensForBaseToken	Private 🔐	🛑	
└	addLiquidity	Private 🔐	🛑	
└	<Receive Ether>	External ❗	💵	NO ❗

ERC20	Implementation	Context, IERC20, IERC20Metadata
└	<Constructor>	Public ❗	🛑	NO ❗
└	name	Public ❗		NO ❗
└	symbol	Public ❗		NO ❗
└	decimals	Public ❗		NO ❗
└	totalSupply	Public ❗		NO ❗
└	balanceOf	Public ❗		NO ❗
└	transfer	Public ❗	🛑	NO ❗
└	allowance	Public ❗		NO ❗
└	approve	Public ❗	🛑	NO ❗
└	transferFrom	Public ❗	🛑	NO ❗
└	increaseAllowance	Public ❗	🛑	NO ❗
└	decreaseAllowance	Public ❗	🛑	NO ❗
└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_spendAllowance	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	

IERC20	Interface	
└	totalSupply	External ❗		NO ❗
└	balanceOf	External ❗		NO ❗
└	transfer	External ❗	🛑	NO ❗
└	allowance	External ❗		NO ❗
└	approve	External ❗	🛑	NO ❗
└	transferFrom	External ❗	🛑	NO ❗

| **IERC20Metadata** | Interface | IERC20 |

CONTRACT ASSESMENT

└	name	External ❗		NO ❗
└	symbol	External ❗		NO ❗
└	decimals	External ❗		NO ❗
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO ❗
└	owner	Public ❗		NO ❗
└	_checkOwner	Internal 🔒		
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_transferOwnership	Internal 🔒	🛑	
SafeERC20	Library			
└	safeTransfer	Internal 🔒	🛑	
└	safeTransferFrom	Internal 🔒	🛑	
└	safeApprove	Internal 🔒	🛑	
└	safeIncreaseAllowance	Internal 🔒	🛑	
└	safeDecreaseAllowance	Internal 🔒	🛑	
└	safePermit	Internal 🔒	🛑	
└	_callOptionalReturn	Private 🔐	🛑	
IERC20Permit	Interface			
└	permit	External ❗	🛑	NO ❗
└	nonces	External ❗		NO ❗
└	DOMAIN_SEPARATOR	External ❗		NO ❗
Address	Library			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionStaticCall	Internal 🔒		
└	functionStaticCall	Internal 🔒		
└	functionDelegateCall	Internal 🔒	🛑	
└	functionDelegateCall	Internal 🔒	🛑	
└	verifyCallResultFromTarget	Internal 🔒		
└	verifyCallResult	Internal 🔒		

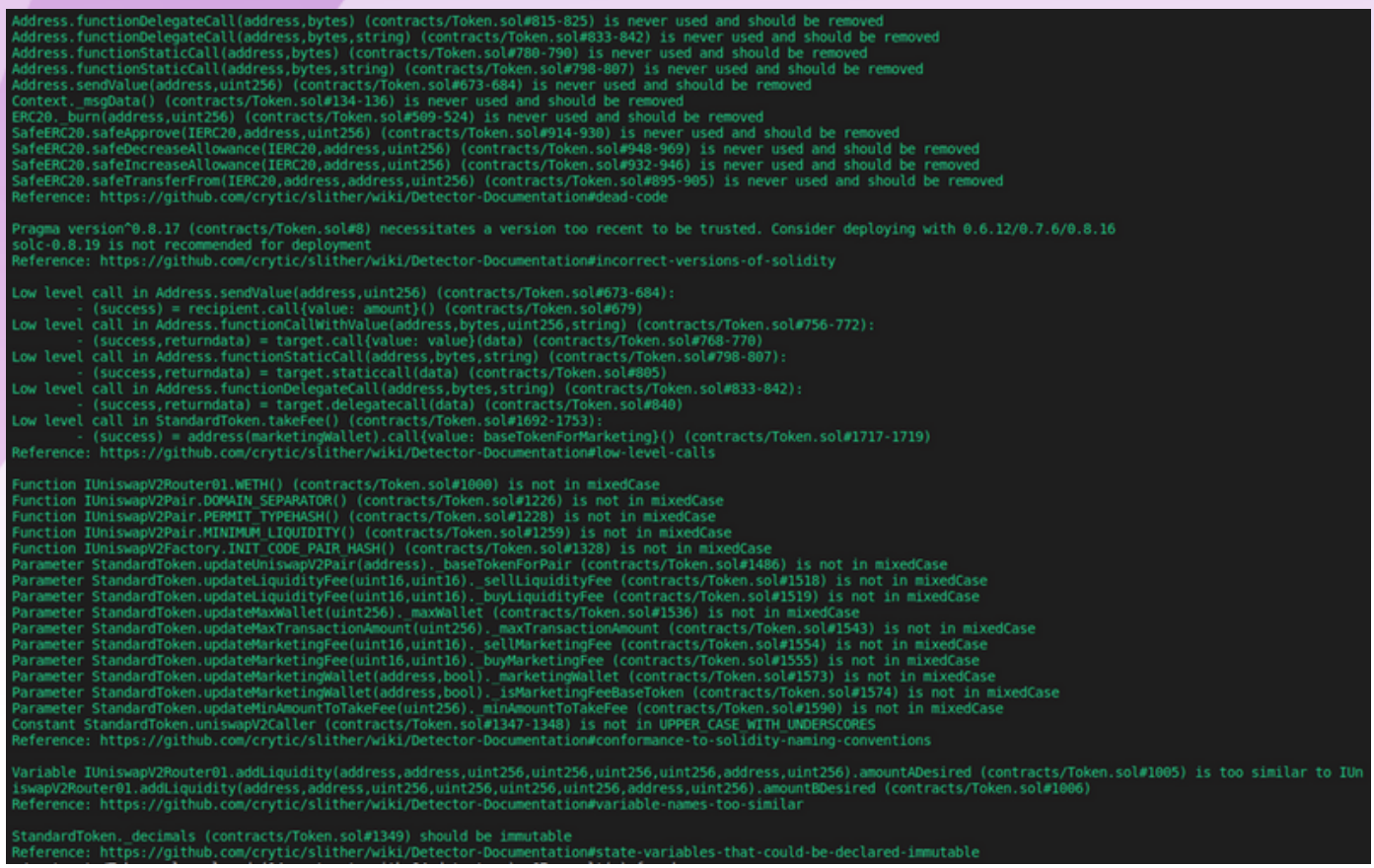
CONTRACT ASSESMENT

| └ | _revert | Private 🔐 | | |

Help Menu

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS



```
Address.functionDelegateCall(address,bytes) (contracts/Token.sol#815-825) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (contracts/Token.sol#833-842) is never used and should be removed
Address.functionStaticCall(address,bytes) (contracts/Token.sol#780-790) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (contracts/Token.sol#798-807) is never used and should be removed
Address.sendValue(address,uint256) (contracts/Token.sol#673-684) is never used and should be removed
Context._msgData() (contracts/Token.sol#134-136) is never used and should be removed
ERC20._burn(address,uint256) (contracts/Token.sol#509-524) is never used and should be removed
SafeERC20.safeApprove(IERC20,address,uint256) (contracts/Token.sol#914-930) is never used and should be removed
SafeERC20.safeDecreaseAllowance(IERC20,address,uint256) (contracts/Token.sol#948-969) is never used and should be removed
SafeERC20.safeIncreaseAllowance(IERC20,address,uint256) (contracts/Token.sol#932-946) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (contracts/Token.sol#895-905) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.17 (contracts/Token.sol#8) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.19 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (contracts/Token.sol#673-684):
        - (success) = recipient.call{value: amount}() (contracts/Token.sol#679)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/Token.sol#756-772):
        - (success,returndata) = target.call{value: value}(data) (contracts/Token.sol#768-770)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/Token.sol#798-807):
        - (success,returndata) = target.staticcall(data) (contracts/Token.sol#805)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/Token.sol#833-842):
        - (success,returndata) = target.delegatecall(data) (contracts/Token.sol#840)
Low level call in StandardToken.takeFee() (contracts/Token.sol#1692-1753):
        - (success) = address(marketingWallet).call{value: baseTokenForMarketing}() (contracts/Token.sol#1717-1719)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function IUniswapV2Router01.WETH() (contracts/Token.sol#1000) is not in mixedCase
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (contracts/Token.sol#1226) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (contracts/Token.sol#1228) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (contracts/Token.sol#1259) is not in mixedCase
Function IUniswapV2Factory.INIT_CODE_PAIR_HASH() (contracts/Token.sol#1328) is not in mixedCase
Parameter StandardToken.updateUniswapV2Pair(address)._baseTokenForPair (contracts/Token.sol#1486) is not in mixedCase
Parameter StandardToken.updateLiquidityFee(uint16,uint16)._sellLiquidityFee (contracts/Token.sol#1518) is not in mixedCase
Parameter StandardToken.updateLiquidityFee(uint16,uint16)._buyLiquidityFee (contracts/Token.sol#1519) is not in mixedCase
Parameter StandardToken.updateMaxWallet(uint256)._maxWallet (contracts/Token.sol#1536) is not in mixedCase
Parameter StandardToken.updateMaxTransactionAmount(uint256)._maxTransactionAmount (contracts/Token.sol#1543) is not in mixedCase
Parameter StandardToken.updateMarketingFee(uint16,uint16)._sellMarketingFee (contracts/Token.sol#1554) is not in mixedCase
Parameter StandardToken.updateMarketingFee(uint16,uint16)._buyMarketingFee (contracts/Token.sol#1555) is not in mixedCase
Parameter StandardToken.updateMarketingWallet(address,bool)._marketingWallet (contracts/Token.sol#1573) is not in mixedCase
Parameter StandardToken.updateMarketingWallet(address,bool)._isMarketingFeeBaseToken (contracts/Token.sol#1574) is not in mixedCase
Parameter StandardToken.updateMinAmountToTakeFee(uint256)._minAmountToTakeFee (contracts/Token.sol#1590) is not in mixedCase
Constant StandardToken.uniswapV2Caller (contracts/Token.sol#1347-1348) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (contracts/Token.sol#1005) is too similar to IUn
iswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/Token.sol#1006)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

StandardToken._decimals (contracts/Token.sol#1349) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
```

Result => A static analysis of contract's source code has been performed using slither,

No major issues were found in the output

FUNCTIONAL TESTING

Router (PCS V2):

0xD99D1c33F9fC3444f8101754aBC46c52416550D1

All the functionalities have been tested, no issues were found

1- Adding liquidity (passed):

https://testnet.bscscan.com/tx/0xa8d9f0588eecc240b9c7fab7b3e1cc599aa1b711abc381903cc43172a1e6739b

2- Buying when excluded (0% tax) (passed):

https://testnet.bscscan.com/tx/0x472a220e01da885f1e3dca38835708c76976aafe9e5d7856cf6e544c293a92e9

3- Selling when excluded (0% tax) (passed):

https://testnet.bscscan.com/tx/0x8fc1cfc3a34fff5e09d702e149abde90696d469e48cb85e998e9235e2bada234

4- Transferring when excluded from fees (0% tax) (passed):

https://testnet.bscscan.com/tx/0xd30f5283c89c63b898ac743de015a8ea8b7eba82cf54a57ebaa8e7e872e72a78

5- Buying when not excluded from fees (up to 0% tax): (passed)

https://testnet.bscscan.com/tx/0x494a72970d3255e88153e244f2ccc756af60eec6329a802b42ab55da98b21f72

6- Selling when not excluded from fees (up to 0% tax) : (passed)

https://testnet.bscscan.com/tx0x5d8f1f884d5fc17ede400b8afa010cb40e20886943b85fc1dab1eff2e9243a16

FUNCTIONAL TESTING

7- Transferring when not excluded from fees (0% tax) (passed):

https://testnet.bscscan.com/tx/0x9554362a05942a2cd5af8680480ab3ecafaad02698a8b712d869cc6bc525a9f9

8- Internal swap (passed):

All fee wallets received BNB + auto liquidity

https://testnet.bscscan.com/tx/0x5d8f1f884d5fc17ede400b8afa010cb40e20886943b85fc1dab1eff2e9243a16

MANUAL TESTING

Centralization - Max amount for buy/transfer/wallet/sells

Severity: High
Function: updateMaxTransactionAmount - updateMaxWallet
Status: Not Resolved

Overview:

The smart contract allows the owner to set maximum limits for buy, transfer, wallet amounts, and sell transactions. The owner can set these limits to as low as 1 wei (0.000000000000000001 PEPA tokens), which could potentially disable buy, transfer, and sell operations for the PEPA token. This high degree of control by the contract owner introduces centralization risks and could impact the token's overall functionality.

```
function updateMaxTransactionAmount(uint256 _maxTransactionAmount) external onlyOwner {
    require(_maxTransactionAmount > 0, "maxTransactionAmount > 0");
    emit UpdateMaxTransactionAmount(_maxTransactionAmount, maxTransactionAmount);
    maxTransactionAmount = _maxTransactionAmount;
}

function updateMaxWallet(uint256 _maxWallet) external onlyOwner {
    require(_maxWallet > 0, "maxWallet > 0");
    emit UpdateMaxWallet(_maxWallet, maxWallet);
    maxWallet = _maxWallet;
}
```

Suggestion

To mitigate this centralization issue, we propose the following options:

1. Renounce Ownership: Consider relinquishing control of the smart contract by renouncing ownership. This would remove the ability for a single entity to manipulate the maximum limits, reducing centralization risks.
2. Multi-signature Wallet: Transfer ownership to a multi-signature wallet. This would require multiple approvals for any changes to the maximum limits, adding an additional layer of security and reducing the centralization risk.

Alleviation:

Owner ship of the contract is renounced,
hence this function can no longer be used by the owner

MANUAL TESTING

Logical – Router can be updated to a new address

Severity: High

function: updateUniswapV2Pair

Status: not resolved

Overview:

The smart contract allows the owner to update the mainRouter which is the contract that is used to add liquidity for the token during an internal swap. Setting mainRouter to a new contract that has some unknowns issues or doesn't support adding liquidity operations, can cause unknown issues for sells (potentially disabling them)

```
function updateUniswapV2Router(address newAddress) public onlyOwner {
    require(
        newAddress != address(mainRouter),
        "The router already has that address"
    );
    emit UpdateUniswapV2Router(newAddress, address(mainRouter));
    mainRouter = IUniswapV2Router02(newAddress);
    address _mainPair = IUniswapV2Factory(mainRouter.factory()).createPair(
        address(this),
        baseTokenForPair
    );
    mainPair = _mainPair;
    _setAutomatedMarketMakerPair(mainPair, true);
}
```

Suggestion

To mitigate this centralization issue, we propose the following options:

1. Renounce Ownership: Consider relinquishing control of the smart contract by renouncing ownership. This would remove the ability for a single entity to manipulate the router, reducing centralization risks.
2. Multi-signature Wallet: Transfer ownership to a multi-signature wallet. This would require multiple approvals for any changes to the mainRouter, adding an additional layer of security and reducing the centralization risk.

Alleviation:

Owner ship of the contract is renounced,
hence this function can no longer be used by the owner

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. 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 Auditace 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 Auditace 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 Auditace team disclaims any liability for the resulting losses.

ABOUT AUDITACE

We specializes in providing thorough and reliable audits for Web3 projects. With a team of experienced professionals, we use cutting-edge technology and rigorous methodologies to evaluate the security and integrity of blockchain systems. We are committed to helping our clients ensure the safety and transparency of their digital assets and transactions.

https://auditace.tech/

https://t.me/Audit_Ace

https://twitter.com/auditace_

https://github.com/Audit-Ace