auditace.tech

AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

RELIQUIDITY PEPE

DATED : 17 June 23'

AUDIT SUMMARY

Project name – RELIQUIDITY PEPE

Date: 17 June, 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	1	1	1	2
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 done on BSC Test network, each test has its transaction has attached to it.

3- Slither : Static Analysis

Testnet Link: all tests were done using this contract, tests are done on BSC Testnet

https://testnet.bscscan.com/token/0x1B7d813B4F4B538e06B67e87326c0e98613c5d5e

Token Information

Token Name : RELIQUIDITY PEPE

Token Symbol: RPEPE

Decimals: 18

Token Supply:500,000

Token Address:
0x2904a4321cE1bA08155c0C12B5564e49a210B11d

Checksum:
b9954e020f1cae91859a5016877e27017e4b2c64

Owner:
0x97681c12dD3A7889cEC0786Bdcb57fA2CeA84D30

TOKEN OVERVIEW

Fees:

Buy Fees: 0-25%

Sell Fees: 0-25%

Transfer Fees: 0-5%

Fees Privilige: Owner

Ownership : Owned

Minting: No mint function

Max Tx: 0.1% - 100% supply

Blacklist: No

Other Priviliges:

-Changing fees

- changing limits

- enabling trades

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

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

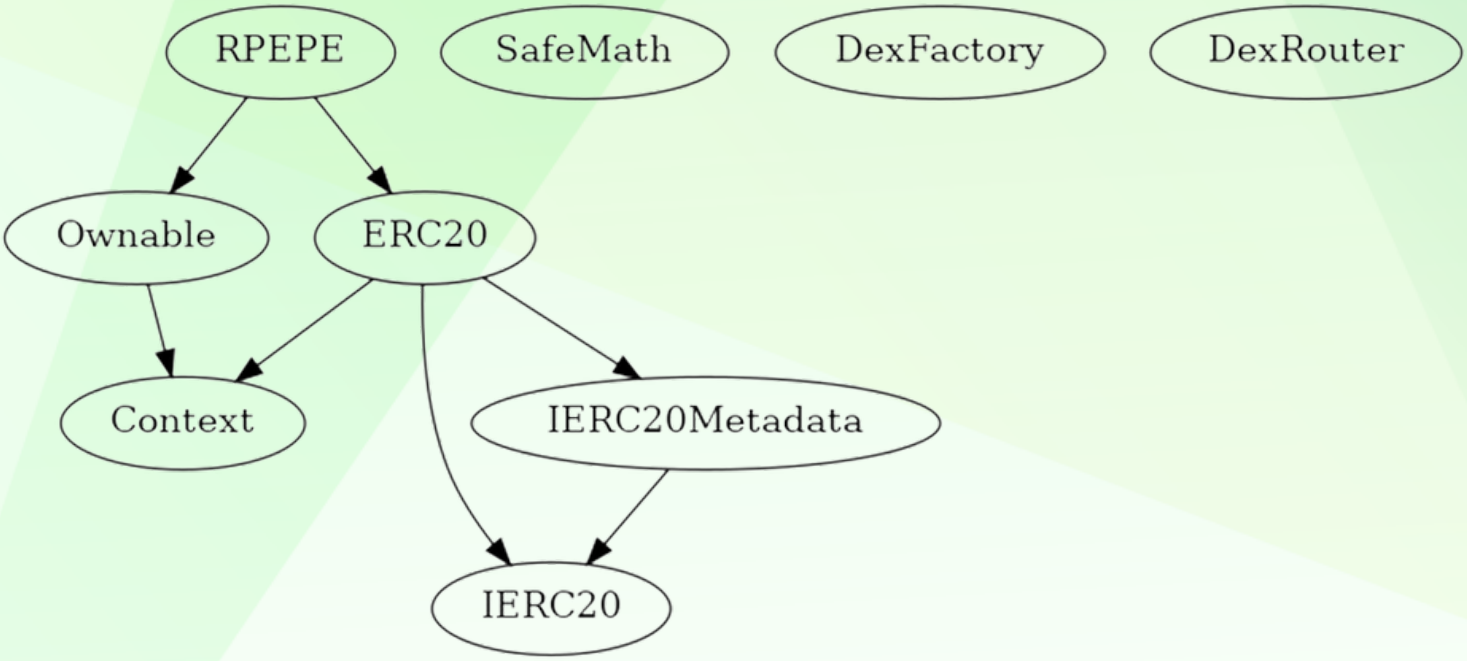
CLASSIFICATION OF RISK

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	1
Medium-Risk	1
Low-Risk	1
Gas Optimization / Suggestions	2

INHERITANCE TREE



POINTS TO NOTE

- Owner is not able to change buy/sell fees over 25% (buy + sell <= 25%)
- Owner is not able to change transfer fees over 5%
- Owner is not able to blacklist an arbitrary address.
- Owner is not able to disable trades
- Owner is able to set max buy/sell/transfer/wallet amount withing a range of 0.1% – 100% of supply
- Owner is not able to mint new tokens
- Owner must enable trades manually for investors

CONTRACT ASSESMENT

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
IERC20	Interface			
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
IERC20Metadata	Interface	IERC20		
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	decimals	External ❗		NO❗
SafeMath	Library			
└	tryAdd	Internal 🔒		
└	trySub	Internal 🔒		
└	tryMul	Internal 🔒		
└	tryDiv	Internal 🔒		
└	tryMod	Internal 🔒		
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	sub	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO❗
└	owner	Public ❗		NO❗
└	_checkOwner	Internal 🔒		
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner

CONTRACT ASSESMENT

└	_transferOwnership	Internal 🔒	🛑		
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 🔒	🛑		
DexFactory	Interface			
└	createPair	External ❗	🛑		NO❗
DexRouter	Interface			
└	factory	External ❗			NO❗
└	WETH	External ❗			NO❗
└	addLiquidityETH	External ❗	💵	NO❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
RPEPE	Implementation	ERC20, Ownable		
└	<Constructor>	Public ❗	🛑	ERC20
└	enableTrading	External ❗	🛑	onlyOwner
└	setmarketingWallet	External ❗	🛑	onlyOwner
└	setreliquidityWallet	External ❗	🛑	onlyOwner
└	setMaxBuy	External ❗	🛑	onlyOwner
└	setMaxSell	External ❗	🛑	onlyOwner
└	setMaxTx	External ❗	🛑	onlyOwner
└	setMaxWallet	External ❗	🛑	onlyOwner
└	setBuyTaxes	External ❗	🛑	onlyOwner

CONTRACT ASSESMENT

└	setSellTaxes	External ❗	🛑	onlyOwner
└	setTransferTaxes	External ❗	🛑	onlyOwner
└	setSwapTokensAtAmount	External ❗	🛑	onlyOwner
└	toggleSwapping	External ❗	🛑	onlyOwner
└	setWhitelistStatus	External ❗	🛑	onlyOwner
└	checkWhitelist	External ❗		NO❗
└	_takeTax	Internal 🔒	🛑	
└	_transfer	Internal 🔒	🛑	
└	internalSwap	Internal 🔒	🛑	
└	swapAndLiquify	Internal 🔒	🛑	
└	swapToETH	Internal 🔒	🛑	
└	addLiquidity	Private 🔐	🛑	
└	withdrawStuckETH	External ❗	🛑	onlyOwner
└	withdrawStuckTokens	External ❗	🛑	onlyOwner
└	<Receive Ether>	External ❗	💵	NO❗

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS

```
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
Pragma version^0.8.0 (contracts/Token.sol#37) allows old versions
Pragma version^0.8.0 (contracts/Token.sol#120) allows old versions
Pragma version^0.8.0 (contracts/Token.sol#153) allows old versions
Pragma version^0.8.0 (contracts/Token.sol#373) allows old versions
Pragma version^0.8.0 (contracts/Token.sol#461) allows old versions
Pragma version^0.8.17 (contracts/Token.sol#836) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.20 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in RPEPE.internalSwap() (contracts/Token.sol#1064-1114):
        - (success) = address(marketingWallet).call{value: (received * marketingPortion) / totalShares}() (contracts/Token.sol#1106)
        - (success) = address(reliquidityWallet).call{value: address(this).balance}() (contracts/Token.sol#1111)
Low level call in RPEPE.withdrawStuckETH() (contracts/Token.sol#1144-1147):
        - (success) = address(msg.sender).call{value: address(this).balance}() (contracts/Token.sol#1145)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function DexRouter.WETH() (contracts/Token.sol#845) is not in mixedCase
Parameter RPEPE.setmarketingWallet(address)._newmarketing (contracts/Token.sol#929) is not in mixedCase
Parameter RPEPE.setreliquidityWallet(address)._newrliquidityWallet (contracts/Token.sol#934) is not in mixedCase
Parameter RPEPE.setMaxBuy(uint256)._mb (contracts/Token.sol#939) is not in mixedCase
Parameter RPEPE.setMaxSell(uint256)._ms (contracts/Token.sol#945) is not in mixedCase
Parameter RPEPE.setMaxTx(uint256)._mt (contracts/Token.sol#951) is not in mixedCase
Parameter RPEPE.setMaxWallet(uint256)._mx (contracts/Token.sol#957) is not in mixedCase
Parameter RPEPE.setBuyTaxes(uint256,uint256,uint256)._lpTax (contracts/Token.sol#963) is not in mixedCase
Parameter RPEPE.setBuyTaxes(uint256,uint256,uint256)._marketingTax (contracts/Token.sol#963) is not in mixedCase
Parameter RPEPE.setBuyTaxes(uint256,uint256,uint256)._rlpTax (contracts/Token.sol#963) is not in mixedCase
Parameter RPEPE.setSellTaxes(uint256,uint256,uint256)._lpTax (contracts/Token.sol#971) is not in mixedCase
Parameter RPEPE.setSellTaxes(uint256,uint256,uint256)._marketingTax (contracts/Token.sol#971) is not in mixedCase
Parameter RPEPE.setSellTaxes(uint256,uint256,uint256)._rlpTax (contracts/Token.sol#971) is not in mixedCase
Parameter RPEPE.setTransferTaxes(uint256,uint256,uint256)._lpTax (contracts/Token.sol#979) is not in mixedCase
Parameter RPEPE.setTransferTaxes(uint256,uint256,uint256)._marketingTax (contracts/Token.sol#979) is not in mixedCase
Parameter RPEPE.setTransferTaxes(uint256,uint256,uint256)._rlpTax (contracts/Token.sol#979) is not in mixedCase
Parameter RPEPE.setSwapTokensAtAmount(uint256)._newAmount (contracts/Token.sol#987) is not in mixedCase
Parameter RPEPE.setWhitelistStatus(address,bool)._wallet (contracts/Token.sol#1000) is not in mixedCase
Parameter RPEPE.setWhitelistStatus(address,bool)._status (contracts/Token.sol#1000) is not in mixedCase
Parameter RPEPE.checkWhitelist(address)._wallet (contracts/Token.sol#1005) is not in mixedCase
Parameter RPEPE.swapAndLiquify(uint256)._amount (contracts/Token.sol#1116) is not in mixedCase
Parameter RPEPE.swapToETH(uint256)._amount (contracts/Token.sol#1127) is not in mixedCase
Parameter RPEPE.addLiquidity(uint256,uint256).ETHAmount (contracts/Token.sol#1137) is not in mixedCase
Parameter RPEPE.withdrawStuckTokens(address).erc20_token (contracts/Token.sol#1149) is not in mixedCase
Constant RPEPE._totalSupply (contracts/Token.sol#872) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

RPEPE.slitherConstructorVariables() (contracts/Token.sol#865-1155) uses literals with too many digits:
        - swapTokensAtAmount = _totalSupply / 100000 (contracts/Token.sol#890)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
```

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/0x1236ea6bbc4f966e4ef1f496ae8bf4cb52aefabe5fc6892ea98b70313aba972a

2- Buying when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x528ba7bf91e76cfb19799c0089e2729dd9acbd673616822d24939996a70e8d15

3- Selling when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0xef56b42a31ca9bb0a026508573e7082351747635e0632d31c79989ecc1fde130

4- Transferring when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x6cf7ae6d50a90b2316c4aea2941c30672758e2e340c47c5d563e220385ee3ed9

5- Buying when not excluded from fees (0-25% tax) (passed):
https://testnet.bscscan.com/tx/0x6227b370816175a0fe3d0f24b1460a6ad0245c34b500a60adca2d30227f74dcd

6- Selling when not excluded from fees (0-25% tax) (passed):
https://testnet.bscscan.com/tx/0xb37ac89d804269b2738f697b96d38973cd7effe30f830afd636c3d9f56b3f1e7

FUNCTIONAL TESTING

7- Transferring when not excluded from fees (0-5% tax) (passed):
https://testnet.bscscan.com/tx/0x1d2212eff88d46cd808c5c8643e207290ac10a499f197c1a1273ac7cfa9ea4ca

8-Internal swap (auto-liquidity and bnb fees) (passed):
https://testnet.bscscan.com/tx/0x1d2212eff88d46cd808c5c8643e207290ac10a499f197c1a1273ac7cfa9ea4ca

ISSUES FOUND

Category: Centralization
Subject: **Centralized control over trading status**
Severity: High
Overview:
The contract owner must enable trades for investors to be able to trade. If trading remain disabled no one would be able to trades their tokens

Code:

```
function enableTrading() external onlyOwner {
    require(!tradingStatus, "trading is already enabled");
    tradingStatus = true;
}
```

ISSUES FOUND

Category: Centralization
Subject: Centralized control over fees and limits
Severity: Medium
Overview:
The contract owner has the ability to set buy, sell, and transfer taxes, as well as maximum buy, sell, transfer, and wallet limits. This centralizes control over fees and limits.
Status: Resolved (fee and limits are within a safe range)
- **Buy + Sell Fees <= 25%**
- **Transfer Fees <= 5%**
- **Limits >= 0.1% of total supply**

Code:
```
function setBuyTaxes( uint256 _lpTax, uint256 _marketingTax, uint256 _rlpTax ) external onlyOwner { ... }
function setSellTaxes( uint256 _lpTax, uint256 _marketingTax, uint256 _rlpTax ) external onlyOwner { ... }
function setTransferTaxes( uint256 _lpTax, uint256 _marketingTax, uint256 _rlpTax ) external onlyOwner { ... }
function setMaxBuy(uint256 _mb) external onlyOwner { ... }
function setMaxSell(uint256 _ms) external onlyOwner { ... }
function setMaxTx(uint256 _mt) external onlyOwner { ... }
function setMaxWallet(uint256 _mx) external onlyOwner { ... }
```

Suggestion:
Consider removing the centralized control over fees and limits or have proper max/min value for each fee or limit.

ISSUES FOUND

Category: Centralization
Subject: **Centralized control over whitelist status**
Severity: Low
Status: not applicable
Overview:
The contract owner has the ability to whitelist or un-whitelist addresses by calling the `**setWhitelistStatus**()` function. This centralizes control over the whitelist status of addresses.

Code:

```
function setWhitelistStatus( address _wallet, bool _status ) external onlyOwner {
    whitelisted[_wallet] = _status;
    emit Whitelist(_wallet, _status);
}
```

Suggestion:
Consider removing the `setWhitelistStatus()` function or implementing a decentralized governance mechanism to control the whitelist status of addresses.

ISSUES FOUND

Category: **Informational**
Subject: Centralized control over swapping and liquidity
Overview:
The contract owner has the ability to enable or disable swapping and liquidity by calling the `**toggleSwapping**()` function. This centralizes control over the swapping and liquidity mechanism.

Code:
```
function toggleSwapping() external onlyOwner {
    swapAndLiquifyEnabled = (swapAndLiquifyEnabled) ? false : true;
}
```

Category: Informational
Subject: Centralized control over marketing and reliquidity wallets
Overview:
The contract owner has the ability to set the marketing and reliquidity wallets by calling the `setmarketingWallet()` and `setreliquidityWallet()` functions. This centralizes control over the wallets receiving the marketing and reliquidity portions of the taxes.

Code:
```
function setmarketingWallet(address _newmarketing) external onlyOwner { ... }
function setreliquidityWallet( address _newrliquidityWallet ) external onlyOwner { ... }
```

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