AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

BarbiePepeX

DATED : 1 Aug 23'

AUDIT SUMMARY

Project name – BarbiePepeX

Date: 1 Aug, 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	1	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/0xA7e06a1bEC030675F13bf50B0e77F56C78a9DaD7

Token Information

Token Name : BarbiePepeX

Token Symbol: BarbiePepeX

Decimals: 18

Token Supply: 1,000,000,000

Token Address:
0xeEA1597264Baa91C5b4b380Ef384e5A8E495d447

Checksum:
0d13ff50475c3fea38371e558f4b13bc5a383542

Owner:
0xa36E787569E115b15B8AE2CAAAe9936C84D8713b
(at time of writing the audit)

Deployer:
0xa36E787569E115b15B8AE2CAAAe9936C84D8713b

TOKEN OVERVIEW

Fees:

Buy Fees: 0-49%

Sell Fees: 0-49%

Transfer Fees: 0%

Fees Privilege: Owner

Ownership: owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Privileges: Initial distribution of the tokens

- modifying fees

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

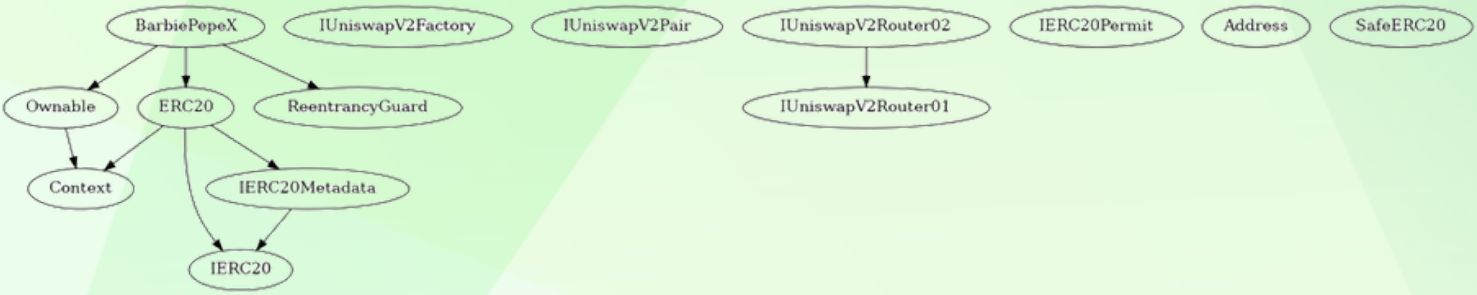
CLASSIFICATION OF RISK

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	1
Low-Risk	0
Gas Optimization / Suggestions	0

INHERITANCE TREE



POINTS TO NOTE

- Owner is able to change current fees within 0-50% for buy and sells fees
- Owner is not able to set buy on transfers
- Owner is not able to blacklist an arbitrary address.
- Owner is not able to mint new tokens
- Owner is not able to set max buy/sell/transfer

CONTRACT ASSESMENT

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
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 ❗
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 ❗
└	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 ❗

CONTRACT ASSESMENT

└	skim	External ❗	🛑	NO ❗
└	sync	External ❗	🛑	NO ❗
└	initialize	External ❗	🛑	NO ❗
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 ❗
IERC20Permit	Interface			
└	permit	External ❗	🛑	NO ❗
└	nonces	External ❗		NO ❗
└	DOMAIN_SEPARATOR	External ❗		NO ❗
Address	Library			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	

CONTRACT ASSESMENT

└	functionCallWithValue	Internal 🔒	🛑	
└	functionStaticCall	Internal 🔒		
└	functionStaticCall	Internal 🔒		
└	functionDelegateCall	Internal 🔒	🛑	
└	functionDelegateCall	Internal 🔒	🛑	
└	verifyCallResultFromTarget	Internal 🔒		
└	verifyCallResult	Internal 🔒		
└	_revert	Private 🔐		
IERC20	Interface			
└	totalSupply	External ❗		NO ❗	
└	balanceOf	External ❗		NO ❗	
└	transfer	External ❗	🛑	NO ❗	
└	allowance	External ❗		NO ❗	
└	approve	External ❗	🛑	NO ❗	
└	transferFrom	External ❗	🛑	NO ❗	
SafeERC20	Library			
└	safeTransfer	Internal 🔒	🛑	
└	safeTransferFrom	Internal 🔒	🛑	
└	safeApprove	Internal 🔒	🛑	
└	safeIncreaseAllowance	Internal 🔒	🛑	
└	safeDecreaseAllowance	Internal 🔒	🛑	
└	safePermit	Internal 🔒	🛑	
└	_callOptionalReturn	Private 🔐	🛑	
IERC20Metadata	Interface	IERC20		
└	name	External ❗		NO ❗	
└	symbol	External ❗		NO ❗	
└	decimals	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 ❗	

CONTRACT ASSESMENT

└	decreaseAllowance	Public ❗	🛑	NO ❗
└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_spendAllowance	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO ❗
└	owner	Public ❗		NO ❗
└	_checkOwner	Internal 🔒		
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_transferOwnership	Internal 🔒	🛑	
ReentrancyGuard	Implementation			
└	<Constructor>	Public ❗	🛑	NO ❗
└	_nonReentrantBefore	Private 🔐	🛑	
└	_nonReentrantAfter	Private 🔐	🛑	
└	_reentrancyGuardEntered	Internal 🔒		
BarbiePepeX	Implementation	ERC20, Ownable, ReentrancyGuard		
└	<Constructor>	Public ❗	🛑	ERC20
└	<Receive Ether>	External ❗	💵	NO ❗
└	<Fallback>	External ❗	💵	NO ❗
└	getRouterAddress	Public ❗		NO ❗
└	claimStuckTokens	External ❗	🛑	onlyOwner
└	excludeFromFees	External ❗	🛑	onlyOwner
└	isExcludedFromFees	Public ❗		NO ❗
└	setAutomatedMarketMakerPair	Public ❗	🛑	onlyOwner
└	isAutomatedMarketMakerPair	Public ❗		NO ❗
└	updateTax	External ❗	🛑	onlyOwner
└	toggleSwapBack	External ❗	🛑	onlyOwner
└	setSwapTokensAtAmount	External ❗	🛑	onlyOwner
└	_transfer	Internal 🔒	🛑	
└	ForceSwapBack	External ❗	🛑	NO ❗
└	_autoswapBack	Internal 🔒	🛑	
└	outBNB	Internal 🔒	🛑	nonReentrant

CONTRACT ASSESMENT

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS

```
Pragma version^0.8.17 (contracts/Token.sol#7) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.21 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#366-374):
        - (success) = recipient.call{value: amount}() (contracts/Token.sol#369)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (contracts/Token.sol#406-419):
        - (success,returndata) = target.call{value: value}(data) (contracts/Token.sol#416)
Low level call in Address.functionStaticCall(address,bytes,string) (contracts/Token.sol#429-437):
        - (success,returndata) = target.staticcall(data) (contracts/Token.sol#434)
Low level call in Address.functionDelegateCall(address,bytes,string) (contracts/Token.sol#451-459):
        - (success,returndata) = target.delegatecall(data) (contracts/Token.sol#456)
Low level call in BarbiePepeX.outBNB(address,uint256) (contracts/Token.sol#1130-1136):
        - (success) = address(_to).call{value: amount}() (contracts/Token.sol#1133)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function IUniswapV2Pair.DOMAIN_SEPARATOR() (contracts/Token.sol#79) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (contracts/Token.sol#81) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (contracts/Token.sol#112) is not in mixedCase
Function IUniswapV2Router01.WETH() (contracts/Token.sol#152) is not in mixedCase
Function IERC20Permit.DOMAIN_SEPARATOR() (contracts/Token.sol#358) is not in mixedCase
Parameter BarbiePepeX.updateTax(uint256,uint256)._buyTax (contracts/Token.sol#1014) is not in mixedCase
Parameter BarbiePepeX.updateTax(uint256,uint256)._sellTax (contracts/Token.sol#1014) is not in mixedCase
Function BarbiePepeX.ForceSwapBack() (contracts/Token.sol#1094-1100) is not in mixedCase
Parameter BarbiePepeX.outBNB(address,uint256)._to (contracts/Token.sol#1130) is not in mixedCase
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#157) is too similar to IUniswapV2Router01.addLiquid
ity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/Token.sol#158)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

BarbiePepeX.marketingWallet (contracts/Token.sol#902) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

BarbiePepeX.marketingWalletShares (contracts/Token.sol#904) should be immutable
BarbiePepeX.setSwapTokensLimit (contracts/Token.sol#907) should be immutable
BarbiePepeX.taxDenominator (contracts/Token.sol#897) should be immutable
BarbiePepeX.uniswapV2Pair (contracts/Token.sol#912) should be immutable
BarbiePepeX.uniswapV2Router (contracts/Token.sol#911) 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

1- Adding liquidity (passed):

https://testnet.bscscan.com/tx/0x44a88b384bd2d559a6e0e9cb61bce83b7603d015ab9f14940046541cdf5fb437

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

https://testnet.bscscan.com/tx/0x3a1f3523692bbbcf3f69d554f7a5b81b7b7985110ed3d107b2dce791898700cf

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

https://testnet.bscscan.com/tx/0x43e6a0242ee433606e13c328a722973204ccc3f83e7b6935aed9cb7d1a14903a

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

https://testnet.bscscan.com/tx/0xe11955ca0296b3b907c3f734118cb15691212c5cef543a5390cad77f89d0f4e6

5- Buying when not excluded from fees (0-49% tax) (passed):

https://testnet.bscscan.com/tx/0x7430ef67ffa5e3087290c7e308aac4133df29e70ec1f53c777f41bed4fd87fc0

6- Selling when not excluded from fees (0-49% tax) (passed):

https://testnet.bscscan.com/tx/0xe8186993c8f3abbf8317bbff1b15b70dc0244b829049d4539dac22240ab264c8

FUNCTIONAL TESTING

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

https://testnet.bscscan.com/tx/0x0c54aed4dd6cf27764d02f7160305c5089763cdc18f1e7b0bc2b4b44c8f49476

8- Internal swap (passed):

https://testnet.bscscan.com/address/0x03269203258f5b50699f0b47d79693e6f222100a#internaltx

Medium Risk

Centralization – Excessive fees

Severity: Medium

function: updateTax

Status: Not Resolved

Overview:

Owner is able to set up to 50% tax on buy and sells seperatly

```
function updateTax(uint256 _buyTax, uint256 _sellTax) external onlyOwner {
  buyTax = _buyTax;
  sellTax = _sellTax;
  totalTax = _buyTax + _sellTax;
  require(buyTax <= 49, "buy Fees cannot exceed 49%");
  require(sellTax <= 49, "buy Fees cannot exceed 49%");
  emit UpdateTax(buyTax, sellTax);
}
```

Suggestion

Set a lower upper bound for maximum amount of buy/sell tax (10% is suggested)

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