AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

Pegasus

DATED : 08 Mar 23'

AUDIT SUMMARY

Project name – Pegasus

Date: 08 March, 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 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/0x328a93d57d0f2ff06f82f5738a563422cb7d50cd#code

Token Information

Token Name : Pegasus

Token Symbol: PEG

Decimals: 18

Token Supply: 1,000,000,000

Token Address:
0x03887C177c18140209C0f93AdC230D4C5515Bf4a

Checksum:
3b33389672bbbb11d901f0852efe7b9895c9a131

Owner:
0xf153B0a6FbB4a091320b3d8a6A7F14a5e4Ef3eB3
(at time of writing the audit)

TOKEN OVERVIEW

Fees:

Buy Fees: 6% max

Sell Fees: 6% max

Transfer Fees: 0%

Fees Privilige: Owner

Ownership : Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Priviliges: including and excluding from fees - changing swap threshold - changing marketing 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

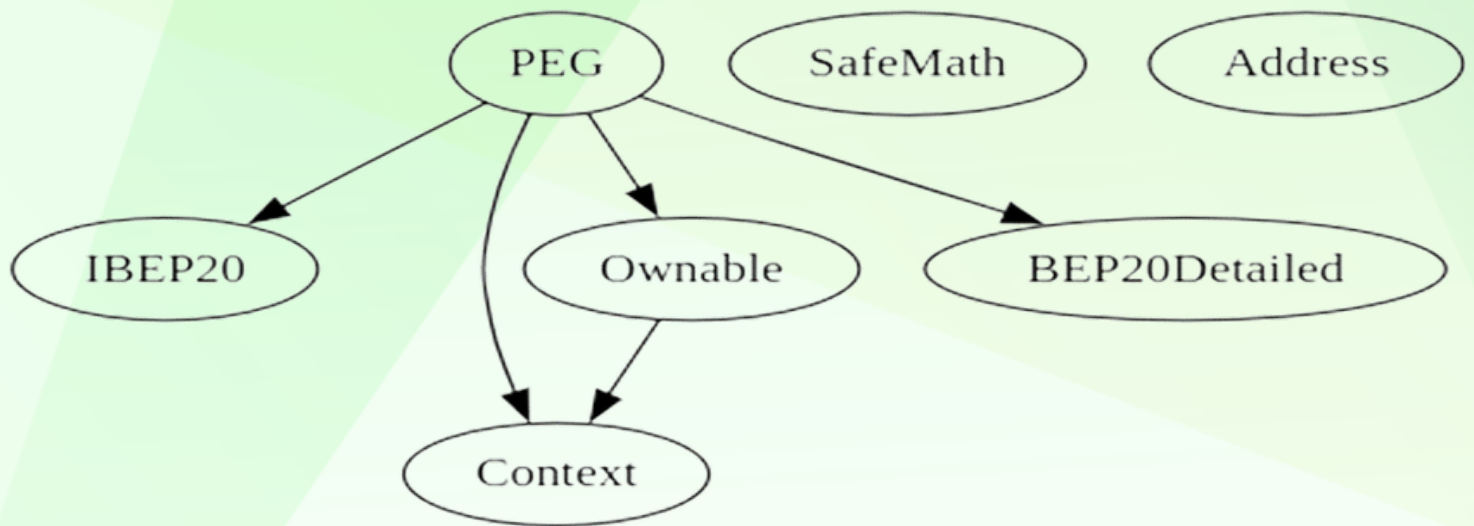| Severity	Found
◆ Critical	0
◆ High-Risk	0
◆ Medium-Risk	0
◆ Low-Risk	0
◆ Gas Optimization / Suggestions	0

INHERITANCE TREE

POINTS TO NOTE

This particular token has the ability to apply a maximum fee of 6% on both buying and selling transactions. These fees are accumulated within the contract until they reach a predetermined threshold, at which point they are converted into BNB and transferred to the marketing wallet. Additionally, there are no restrictions on the maximum amount an investor can trade, nor are there any limits on buy/sell/transfer amounts.

Furthermore, the token has obtained a Pinksale Safu badge from Intefi, indicating that its structure meets the necessary criteria for security and safety.

- **Owner is not able to set buy tax higher than 6%**
- **Owner is not able to set sell tax higher than 6%**
- **Owner is not able to set max buy/sell/transfer/hold amount**
- **Owner is not able to blacklist an arbitrary wallet**
- **Owner is not able to disable trades**
- **Owner is not able to mint new tokens**

CONTRACT ASSESMENT
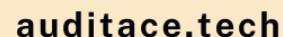
Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
IBEP20	Interface			
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
SafeMath	Library			
└	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 🔒		
Context	Implementation			
└	<Constructor>	Public ❗	🛑	NO❗
└	_msgSender	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO❗
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
BEP20Detailed	Implementation			
└	<Constructor>	Public ❗	🛑	NO❗
└	name	Public ❗		NO❗
└	symbol	Public ❗		NO❗
└	decimals	Public ❗		NO❗

CONTRACT ASSESMENT

Address	Library				
└	isContract	Internal 🔒			
SafeBEP20	Library				
└	safeTransfer	Internal 🔒	🛑		
└	safeTransferFrom	Internal 🔒	🛑		
└	safeApprove	Internal 🔒	🛑		
└	callOptionalReturn	Private 🔏	🛑		
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❗	

CONTRACT ASSESMENT

└	mint	External ❗	🛑	NO❗
└	burn	External ❗	🛑	NO❗
└	swap	External ❗	🛑	NO❗
└	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❗
PEG	Implementation	Context, Ownable, IBEP20, BEP20Detailed		
└	<Constructor>	Public ❗	🛑	BEP20Detailed
└	totalSupply	Public ❗		NO❗
└	balanceOf	Public ❗		NO❗
└	transfer	Public ❗	🛑	NO❗
└	allowance	Public ❗		NO❗

CONTRACT ASSESMENT

└	approve	Public ❗	🛑	NO❗
└	transferFrom	Public ❗	🛑	NO❗
└	increaseAllowance	Public ❗	🛑	NO❗
└	decreaseAllowance	Public ❗	🛑	NO❗
└	setBuyMarketingFeePercent	External ❗	🛑	onlyOwner
└	setSellMarketingFeePercent	External ❗	🛑	onlyOwner
└	setMarketingAddress	External ❗	🛑	onlyOwner
└	setSwapAndLiquifyEnabled	Public ❗	🛑	onlyOwner
└	changeNumTokensSellToFee	External ❗	🛑	onlyOwner
└	excludeFromFee	Public ❗	🛑	onlyOwner
└	includeInFee	Public ❗	🛑	onlyOwner
└	<Receive Ether>	External ❗	💵	NO❗
└	_transfer	Internal 🔒	🛑	
└	swapAndLiquify	Private 🔐	🛑	lockTheSwap
└	swapTokensForEth	Private 🔐	🛑	
└	_approve	Internal 🔒	🛑	

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2

Reentrancy in SHIKOKUCEO._transfer(address,address,uint256) (contracts/Token.sol#463-487):
        External calls:
        - swapAndLiquify() (contracts/Token.sol#480)
                - (success) = recipient.call{value: amount}() (contracts/Token.sol#123)
                - router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (contracts/Token.sol#535-541)
                - address(marketingWallet).sendValue(deltaBalance) (contracts/Token.sol#522)
        External calls sending eth:
        - swapAndLiquify() (contracts/Token.sol#480)
                - (success) = recipient.call{value: amount}() (contracts/Token.sol#123)
        Event emitted after the call(s):
        - Transfer(sender,recipient,s.tTransferAmount) (contracts/Token.sol#513)
                - _tokenTransfer(from,to,amount,takeFee) (contracts/Token.sol#486)
Reentrancy in SHIKOKUCEO.transferFrom(address,address,uint256) (contracts/Token.sol#250-265):
        External calls:
        - _transfer(sender,recipient,amount) (contracts/Token.sol#255)
                - (success) = recipient.call{value: amount}() (contracts/Token.sol#123)
                - router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (contracts/Token.sol#535-541)
                - address(marketingWallet).sendValue(deltaBalance) (contracts/Token.sol#522)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (contracts/Token.sol#255)
                - (success) = recipient.call{value: amount}() (contracts/Token.sol#123)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (contracts/Token.sol#460)
                - _approve(sender,_msgSender(),currentAllowance - amount) (contracts/Token.sol#262)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

SHIKOKUCEO.includeInReward(address) (contracts/Token.sol#340-351) has costly operations inside a loop:
        - _excluded.pop() (contracts/Token.sol#347)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop

Context._msgData() (contracts/Token.sol#45-48) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

SHIKOKUCEO._rTotal (contracts/Token.sol#151) is set pre-construction with a non-constant function or state variable:
        - (MAX - (MAX % _tTotal))
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#function-initializing-state

Pragma version^0.8.17 (contracts/Token.sol#6) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.18 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#117-128):
        - (success) = recipient.call{value: amount}() (contracts/Token.sol#123)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
```

**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/0x35d0a21cc6af7202b48abc8e161
dca2aae23d1ac70ad0d3fef549593f01da67b

2- Buying when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0xc3504e04962f883654f63b29c1
55a5623e266c45dcb2bc7b2c65e0fb8d4863be

3- Selling when excluded from fees(0% tax) (passed):
https://testnet.bscscan.com/tx/0x0c80af63c3c007ba8a2f2e4121e
b1a303db0ceaefb24c3826003f75f31c2a668

4- Transferring when excluded from fees(0% tax) (passed):
https://testnet.bscscan.com/tx/0x11d240b6d0b765c34d5e177add
719b20eb046ff9c8050a1a01450a61f78a47f9

FUNCTIONAL TESTING

5- Buying when not excluded from fees (6% tax) (passed):
https://testnet.bscscan.com/tx/0xec4723a5767806390e940ba77e
cd3a5b67257a91566d3add5a5361e199072630

6- Selling when not excluded from fees (6% tax) (passed):
https://testnet.bscscan.com/tx/0x7fe22c379e4b49f3fdc0459f196
e6c211c5f9519e45c806b9181a6ebf5d37446

7- Transferring when not excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0x20281b59cd0bf77d21ef6636ecb
93447ebfbb1dea1bdd8ec68585df53c6f853c

8-Internal swap (passed):
marketing wallet received ETH
https://testnet.bscscan.com/address/0x7980af1494b801895c45b
84533596f3f58df152f#internaltx

MANUAL TESTING

No Issues Found

Social Media Overview

Here are the Social Media Accounts of

Pegasus

https://t.me/PegasusDEX

https://twitter.com/PegasusDEX

https://pegasusdex.exchange

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