auditace.tech

AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

PUG LIFE

DATED : 21 October 23'

MANUAL TESTING

Centralization – Enabling Trades

Severity: High

function: EnableTrading

Status: Open

Overview:

The enableTrading function permits only the contract owner to activate trading capabilities. Until this function is executed, no investors can buy, sell, or transfer their tokens. This places a high degree of control and centralization in the hands of the contract owner.

```
function EnableTrading() external onlyOwner {
    require(!tradingEnabled, "Cannot re-enable trading");
    tradingEnabled = true;
    swapEnabled = true;
    genesis_block = block.number;
}
```

Suggestion

To reduce centralization and potential manipulation, consider one of the following approaches:

1.Automatically enable trading after a specified condition, such as the completion of a presale, is met.

2.If manual activation is still desired, consider transferring the ownership of the contract to a trustworthy, third-party entity like a certified "PinkSale Safu" developer. This can provide investors with more confidence in the eventual activation of trading capabilities, mitigating concerns of potential bad faith actions by the original owner

AUDIT SUMMARY

Project name – PUG LIFE

Date: 21 October 2023

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

Audit Status: **Passed with high risk**

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	1	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/0xf0A2f58209bb850A76A6c07ae5Ab89c09DE19A94

Token Information

Token Address :
0x9b0E205EED3c6f0597453A8E061F2d05902Dcc3E

Name: PUG LIFE

Symbol: PUGLIFE

Decimals: 9

Network: Binance smart chain

Token Type: BEP20

Owner: 0x6D771c1745AD3a3dB8C98FdAf0e04D3Bb27Ac19A

Deployer:
0x6D771c1745AD3a3dB8C98FdAf0e04D3Bb27Ac19A

Token Supply: 100,000,000

Checksum:
1666029b29a5f1ae543a23971ebc1e066fc0f1b5

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/0xf0A2f58209bb850A76A6c07ae5Ab89c09DE19A94

TOKEN OVERVIEW

buy fee: 4%

Sell fee: 4%

transfer fee: 4%

Fee Privilege: Owner

Ownership: Owned

Minting: None

Max Tx: No

Blacklist: No

Other Privileges:

- Initial distribution of the tokens

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

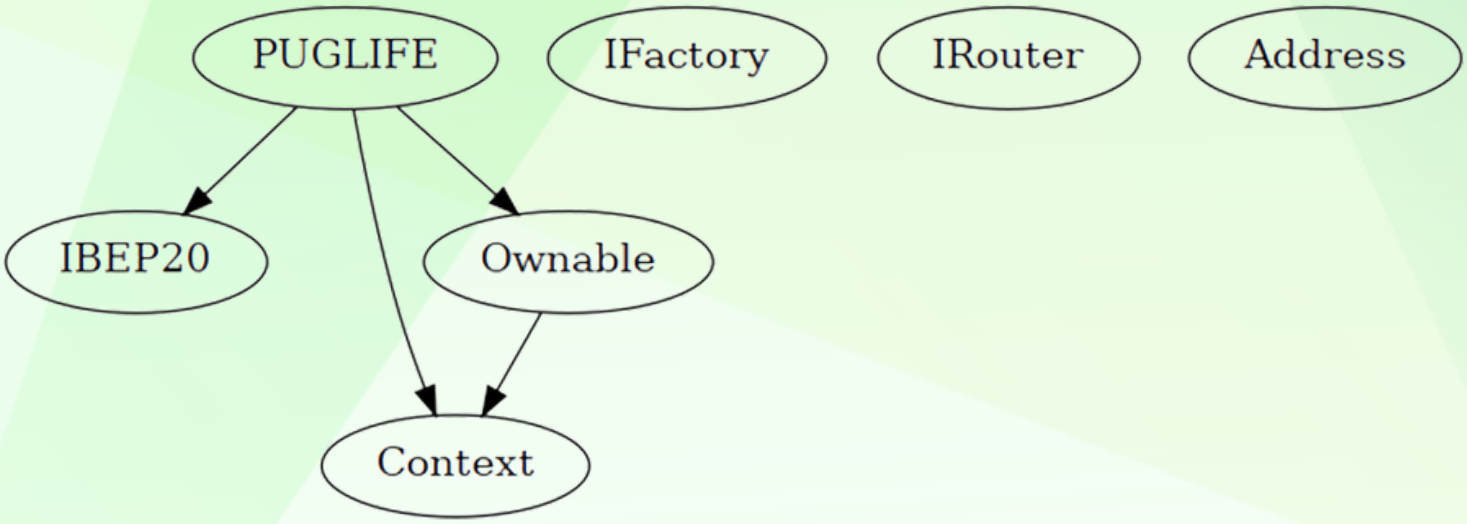
CLASSIFICATION OF RISK

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	0
Low-Risk	0
Gas Optimization / Suggestions	0

INHERITANCE TREE



POINTS TO NOTE

- Owner is not able to adjust buy/sell/transfer fees (3% each)

- Owner is not able to blacklist an arbitrary wallet

- Owner is not able to disable trades

- Owner is not able to mint new tokens

- Owner must enable trades manually (trades already enabled)

STATIC ANALYSIS

```
INFO:Detectors:
PUGLIFE._rTotal (contracts/Token.sol#190) 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
INFO:Detectors:
Pragma version^0.8.17 (contracts/Token.sol#25) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (contracts/Token.sol#144-155):
        - (success) = recipient.call{value: amount}() (contracts/Token.sol#150)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IRouter.WETH() (contracts/Token.sol#120) is not in mixedCase
Struct PUGLIFE.valuesFromGetValues (contracts/Token.sol#227-241) is not in CapWords
Function PUGLIFE.EnableTrading() (contracts/Token.sol#395-400) is not in mixedCase
Parameter PUGLIFE.updatedeadline(uint256)._deadline (contracts/Token.sol#402) is not in mixedCase
Parameter PUGLIFE.updateSwapEnabled(bool)._enabled (contracts/Token.sol#822) is not in mixedCase
Parameter PUGLIFE.rescueAnyBEP20Tokens(address,address,uint256)._tokenAddr (contracts/Token.sol#834) is not in mixedCase
Parameter PUGLIFE.rescueAnyBEP20Tokens(address,address,uint256)._to (contracts/Token.sol#835) is not in mixedCase
Parameter PUGLIFE.rescueAnyBEP20Tokens(address,address,uint256)._amount (contracts/Token.sol#836) is not in mixedCase
Constant PUGLIFE._decimals (contracts/Token.sol#186) is not in UPPER_CASE_WITH_UNDERSCORES
Variable PUGLIFE.genesis_block (contracts/Token.sol#194) is not in mixedCase
Constant PUGLIFE._name (contracts/Token.sol#202) is not in UPPER_CASE_WITH_UNDERSCORES
Constant PUGLIFE._symbol (contracts/Token.sol#203) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (contracts/Token.sol#65)" inContext (contracts/Token.sol#59-68)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
PUGLIFE.updateSwapTokensAtAmount(uint256) (contracts/Token.sol#810-820) uses literals with too many digits:
        - require(bool,string)(amount <= 1000000,Cannot set swap threshold amount higher than 1% of tokens) (contracts/Token.sol#811-814)
PUGLIFE.updateSwapTokensAtAmount(uint256) (contracts/Token.sol#810-820) uses literals with too many digits:
        - require(bool,string)(amount >= (totalSupply() / 100000) * 10 ** decimals(),New swap threshold must be higher than 0.00001% of total supply) (contracts/Token.sol#815-818)
PUGLIFE.slitherConstructorVariables() (contracts/Token.sol#165-851) uses literals with too many digits:
        - _tTotal = 100000000 * 10 ** _decimals (contracts/Token.sol#189)
PUGLIFE.slitherConstructorVariables() (contracts/Token.sol#165-851) uses literals with too many digits:
        - swapTokensAtAmount = 100000 * 10 ** 9 (contracts/Token.sol#192)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
PUGLIFE._lastSell (contracts/Token.sol#181) is never used in PUGLIFE (contracts/Token.sol#165-851)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable
INFO:Detectors:
Loop condition `i < _excluded.length` (contracts/Token.sol#607) should use cached array length instead of referencing `length` member of the storage array.
 Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cache-array-length
INFO:Detectors:
PUGLIFE._tTotal (contracts/Token.sol#189) should be constant
PUGLIFE.deadWallet (contracts/Token.sol#197) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
PUGLIFE.pair (contracts/Token.sol#184) should be immutable
PUGLIFE.router (contracts/Token.sol#183) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
INFO:Slither:./contracts/Token.sol analyzed (7 contracts with 88 detectors), 49 result(s) found
```

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

No major issues were found in the output

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 ❗
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO ❗
└	owner	Public ❗		NO ❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_setOwner	Private 🔐	🛑	
IFactory	Interface			
└	createPair	External ❗	🛑	NO ❗
IRouter	Interface			
└	factory	External ❗		NO ❗
└	WETH	External ❗		NO ❗
└	addLiquidityETH	External ❗	💵	NO ❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
Address	Library			
└	sendValue	Internal 🔒	🛑	

CONTRACT ASSESMENT

PUGLIFE	Implementation	Context, IBEP20, Ownable		
└	<Constructor>	Public ❗	🛑	NO ❗
└	name	Public ❗		NO ❗
└	symbol	Public ❗		NO ❗
└	decimals	Public ❗		NO ❗
└	totalSupply	Public ❗		NO ❗
└	balanceOf	Public ❗		NO ❗
└	allowance	Public ❗		NO ❗
└	approve	Public ❗	🛑	NO ❗
└	transferFrom	Public ❗	🛑	NO ❗
└	increaseAllowance	Public ❗	🛑	NO ❗
└	decreaseAllowance	Public ❗	🛑	NO ❗
└	transfer	Public ❗	🛑	NO ❗
└	isExcludedFromReward	Public ❗		NO ❗
└	reflectionFromToken	Public ❗		NO ❗
└	EnableTrading	External ❗	🛑	onlyOwner
└	updatedeadline	External ❗	🛑	onlyOwner
└	tokenFromReflection	Public ❗		NO ❗
└	excludeFromReward	Public ❗	🛑	onlyOwner
└	includeInReward	External ❗	🛑	onlyOwner
└	excludeFromFee	Public ❗	🛑	onlyOwner
└	includeInFee	Public ❗	🛑	onlyOwner
└	isExcludedFromFee	Public ❗		NO ❗
└	_reflectRfi	Private 🔐	🛑	
└	_takeLiquidity	Private 🔐	🛑	
└	_takeMarketing	Private 🔐	🛑	
└	_takeOps	Private 🔐	🛑	
└	_takeDev	Private 🔐	🛑	
└	_getValues	Private 🔐		
└	_getTValues	Private 🔐		
└	_getRValues1	Private 🔐		
└	_getRValues2	Private 🔐		
└	_getRate	Private 🔐		
└	_getCurrentSupply	Private 🔐		

CONTRACT ASSESMENT

└	_approve	Private 🔐	🛑	
└	_transfer	Private 🔐	🛑	
└	_tokenTransfer	Private 🔐	🛑	
└	swapAndLiquify	Private 🔐	🛑	lockTheSwap
└	addLiquidity	Private 🔐	🛑	
└	swapTokensForBNB	Private 🔐	🛑	
└	bulkExcludeFee	External ❗	🛑	onlyOwner
└	updateMarketingWallet	External ❗	🛑	onlyOwner
└	updateDevWallet	External ❗	🛑	onlyOwner
└	updateOpsWallet	External ❗	🛑	onlyOwner
└	updateSwapTokensAtAmount	External ❗	🛑	onlyOwner
└	updateSwapEnabled	External ❗	🛑	onlyOwner
└	rescueBNB	External ❗	🛑	onlyOwner
└	rescueAnyBEP20Tokens	Public ❗	🛑	onlyOwner
└	<Receive Ether>	External ❗	💵	NO ❗

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

FUNCTIONAL TESTING

1- Adding liquidity (passed):
https://testnet.bscscan.com/tx/0x6e637b500d8dedc8ffdeb5b51c5a5b203b017ef9f271008146965b4c209d4efe

2- Buying when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x7330a73cc450a126dda974097dbe9fe9b42f02922d5b77ba67d9cabdb729a796

3- Selling when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x0a77e8b65d212a6ba0377dfd4fe619239f030ea3ee18bd713de8ddc059885013

4- Transferring when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0x0a77e8b65d212a6ba0377dfd4fe619239f030ea3ee18bd713de8ddc059885013

5- Buying when not excluded from fees (tax 4%) (passed):
https://testnet.bscscan.com/tx/0x1e11ea7d29ff1fa3a3c4d2429fbb274f84c7fe19fae32ca65c8304b1c28e70c4

6- Selling when not excluded from fees (tax 4%) (passed):
https://testnet.bscscan.com/tx/0xc823f234d704a48aa6a1be4d9b90999bd5a8fb7011a5efad105336d1d8cc9980

7- Transferring when not excluded from fees (4% tax) (passed):
https://testnet.bscscan.com/tx/0xcf5d573fc5988a8a5b275e2b23927dd89059234eeea34b50178d40a0ee35b3e4

7- Internal swap (Marketing BNB) (passed):
https://testnet.bscscan.com/tx/0xc823f234d704a48aa6a1be4d9b90999bd5a8fb7011a5efad105336d1d8cc9980

MANUAL TESTING

Centralization – Enabling Trades

Severity: High

function: EnableTrading

Status: Open

Overview:

The enableTrading function permits only the contract owner to activate trading capabilities. Until this function is executed, no investors can buy, sell, or transfer their tokens. This places a high degree of control and centralization in the hands of the contract owner.

```
function EnableTrading() external onlyOwner {
    require(!tradingEnabled, "Cannot re-enable trading");
    tradingEnabled = true;
    swapEnabled = true;
    genesis_block = block.number;
}
```

Suggestion

To reduce centralization and potential manipulation, consider one of the following approaches:

1.Automatically enable trading after a specified condition, such as the completion of a presale, is met.

2.If manual activation is still desired, consider transferring the ownership of the contract to a trustworthy, third-party entity like a certified "PinkSale Safu" developer. This can provide investors with more confidence in the eventual activation of trading capabilities, mitigating concerns of potential bad faith actions by the original 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