auditace.tech

AuditAce

FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

Decentratool

DATED : 6 MAY 23'

AUDIT SUMMARY

Project name – Decentratool

Date: 6 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	1
Acknowledged	0	0	0	0	0
Resolved	0	1	0	0	0

USED TOOLS

Tools:

1.Manual Review: The code has undergone a line-by-line review by the **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.

Token Information

Name : Decentratool

Symbol : DTOOLS

Decimals: 18

Network: Binance smart chain

Token Type: BEP20

Token Address :
0x25acE7DC3c24ef631B5b4e0B56783562a3D54517

Owner:
0xc8a3323975d80b8C6EfDbb9DbfbF42471aF86fCB

Deployer:
0xc8a3323975d80b8C6EfDbb9DbfbF42471aF86fCB

Token Information

Fees:

Buy Fees: Up to 8%

Sell Fees: Up to 8%

Transfer Fees: Up to 8%

Fees Privilige: Owner

Ownership : Owned

Minting: None

Max Tx Amount/ Max Wallet Amount: Yes

Blacklist: No

Other Priviliges: Including or excluding from fees - changing swap threshold - enabling /disabling tax

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.

auditace.tech

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

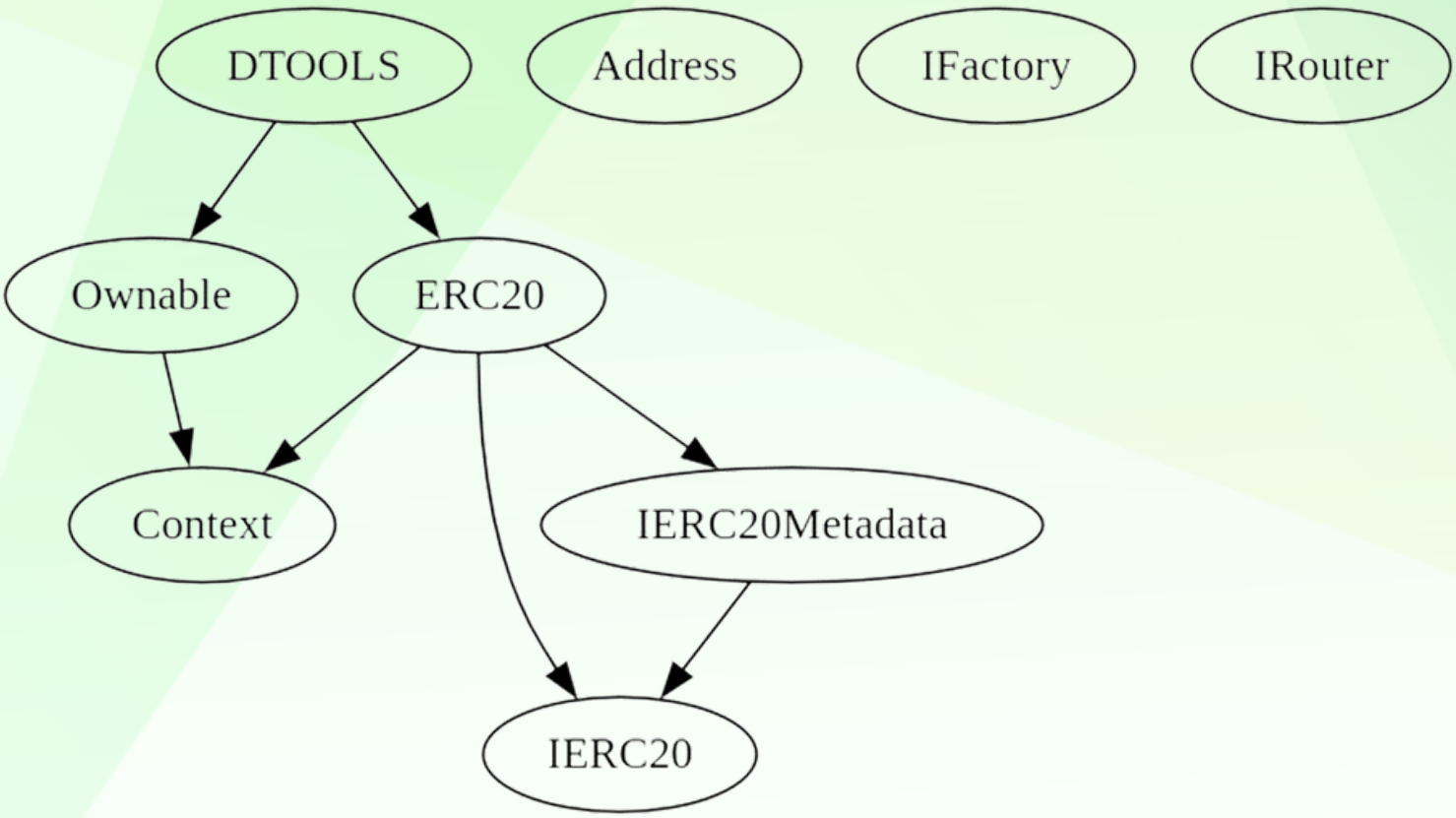
CLASSIFICATION OF RISK

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	1
Medium-Risk	0
Low-Risk	0
Gas Optimization / Suggestions	1

INHERITANCE TREE



POINTS TO NOTE

- Owner is not able to set set buy/sell/transfer tax more than 8% each
- Owner is not able to set a max buy/transfer/wallet/sell 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
- Owner must enable trades for holders to be able to trade

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 ❗
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 🔓	🛑	
└	_tokengeneration	Internal 🔓	🛑	
└	_approve	Internal 🔓	🛑	
Address	Library			
└	sendValue	Internal 🔓	🛑	
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO ❗

CONTRACT ASSESMENT

└	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 ❗
DTOOLS	Implementation	ERC20, Ownable		
└	<Constructor>	Public ❗	🛑	ERC20
└	approve	Public ❗	🛑	NO ❗
└	transferFrom	Public ❗	🛑	NO ❗
└	increaseAllowance	Public ❗	🛑	NO ❗
└	decreaseAllowance	Public ❗	🛑	NO ❗
└	transfer	Public ❗	🛑	NO ❗
└	_transfer	Internal 🔒	🛑	
└	Liquify	Private 🔐	🛑	lockTheSwap
└	swapTokensForETH	Private 🔐	🛑	
└	addLiquidity	Private 🔐	🛑	
└	updateLiquidityProvide	External ❗	🛑	onlyOwner
└	updateLiquidityTreshhold	External ❗	🛑	onlyOwner
└	EnableTrading	External ❗	🛑	onlyOwner
└	UpdateZeroBuyTax	External ❗	🛑	onlyOwner
└	UpdateZeroSellTax	External ❗	🛑	onlyOwner
└	SetBuyTax	External ❗	🛑	onlyOwner
└	SetSellTax	External ❗	🛑	onlyOwner
└	UpdateTxTax	External ❗	🛑	onlyOwner
└	updatedeadline	External ❗	🛑	onlyOwner
└	updateMarketingWallet	External ❗	🛑	onlyOwner
└	updateDevWallet	External ❗	🛑	onlyOwner
└	updateExemptFee	External ❗	🛑	onlyOwner
└	bulkExemptFee	External ❗	🛑	onlyOwner
└	rescueBNB	External ❗	🛑	onlyOwner
└	rescueBEP20	External ❗	🛑	onlyOwner
└	<Receive Ether>	External ❗	💵	NO ❗

CONTRACT ASSESMENT

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS

```
Reentrancy in DTOOLS.transferFrom(address,address,uint256) (contracts/Token.sol#502-516):
        External calls:
        - _transfer(sender,recipient,amount) (contracts/Token.sol#507)
                - router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,deadWallet,block.timestamp) (contracts/Token.sol#677-684)
                - (success) = recipient.call{value: amount}() (contracts/Token.sol#352)
                - router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (contracts/Token.sol#663-669)
                - address(marketingWallet).sendValue(marketingAmt) (contracts/Token.sol#644)
                - address(devWallet).sendValue(devAmt) (contracts/Token.sol#649)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (contracts/Token.sol#507)
                - router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,deadWallet,block.timestamp) (contracts/Token.sol#677-684)
                - (success) = recipient.call{value: amount}() (contracts/Token.sol#352)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (contracts/Token.sol#341)
                - _approve(sender,_msgSender(),currentAllowance - amount) (contracts/Token.sol#513)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Context._msgData() (contracts/Token.sol#22-25) 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#15) 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#346-357):
        - (success) = recipient.call{value: amount}() (contracts/Token.sol#352)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Variable ERC20._balances (contracts/Token.sol#78) is not in mixedCase
Variable ERC20._allowances (contracts/Token.sol#80) is not in mixedCase
Function IRouter.WETH() (contracts/Token.sol#410) is not in mixedCase
Function DTOOLS.Liquify(uint256,DTOOLS.Taxes) (contracts/Token.sol#608-652) is not in mixedCase
Parameter DTOOLS.updateLiquidityTreshhold(uint256).new_amount (contracts/Token.sol#691) is not in mixedCase
Function DTOOLS.EnableTrading() (contracts/Token.sol#703-708) is not in mixedCase
Function DTOOLS.UpdateZeroBuyTax() (contracts/Token.sol#710-712) is not in mixedCase
Function DTOOLS.UpdateZeroSellTax() (contracts/Token.sol#714-716) is not in mixedCase
Function DTOOLS.SetBuyTax() (contracts/Token.sol#718-720) is not in mixedCase
Function DTOOLS.SetSellTax() (contracts/Token.sol#722-724) is not in mixedCase
Function DTOOLS.UpdateTxTax() (contracts/Token.sol#726-729) is not in mixedCase
Parameter DTOOLS.updatedeadline(uint256)._deadline (contracts/Token.sol#731) is not in mixedCase
Parameter DTOOLS.updateExemptFee(address,bool)._address (contracts/Token.sol#747) is not in mixedCase
Variable DTOOLS.genesis_block (contracts/Token.sol#445) is not in mixedCase
Constant DTOOLS.deadWallet (contracts/Token.sol#451-452) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (contracts/Token.sol#23)" inContext (contracts/Token.sol#17-26)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

DTOOLS.launchtax (contracts/Token.sol#447) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

DTOOLS.pair (contracts/Token.sol#437) should be immutable
DTOOLS.router (contracts/Token.sol#436) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
```

FUNCTIONAL TESTING

1- Adding liquidity (passed):
https://testnet.bscscan.com/tx/0x06f0595856f96f37d74fbf983d1d6f8e6a345ed721c183d66db3b31ac78caa18

2- Buying when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x312b35411187e615e3ddd941993644b1e5c84be2fc5bba68abc0cc476276ba9e

3- Selling when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0xa83c23ad9ecb9391f1e84f584d20ec79008c5d25c66c452d6a50379f12825b2f

4- Transferring when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0x5d702163c5b9a207a280e1653d9b4c57a52b1a68133e9798edc3337f3982deb1

5- Buying when not excluded from fees (upto 8% tax) (passed):
https://testnet.bscscan.com/tx/0x6d9bf06582e5f5c2566c15c042967a053699bd4ffb79585180a572d1d48f418d

6- Selling when not excluded from fees (upto 8% tax) (passed):
https://testnet.bscscan.com/tx/0xe2cb8473162071bdf06b9c23d949fb63424e7a7a08433975be1db469316b6339

7- Transferring when not excluded from fees (upto 8% tax) (passed):
https://testnet.bscscan.com/tx/0xffaf7bb46f785bd3bfbcda0ec99d9004a86d1dbdce2061bec16e5753a635bd14

FUNCTIONAL TESTING

7- Internal swap (fee wallets received BNB) (passed):
https://testnet.bscscan.com/address/0x980d0c76baeac720e610054939211d3a15ec98b8

FUNCTIONAL TESTING

Centralization – Trades must be enabled

Severity: High

function: EnableTrading

Status: Resolved (Contract is owned by Pinksale safu developer)

Overview:

The smart contract owner must enable trades for holders. If trading remain disabled, no one would be able to buy/sell/transfer tokens.

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

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.
3. Transfer ownership to a trusted and valid 3rd party in order to guarantee enabling of the trades **(applied)**

FUNCTIONAL TESTING

Informational – Redundant code

Status: Not Resolved

Overview:

Auto-liquidty feature of the contract is never used (0% liqudity tax) hence its suggested to remove auto-liquidity code.

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