AuditAce

FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

DONALDINU

DATED : 8 APR 23'

AUDIT SUMMARY

Project name – DONALDINU

Date: 8 April, 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	0	0	0
Acknowledged	0	0	0	0	0
Resolved	0	0	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 : Donald Inu

Symbol : DONALDINU

Decimals: 9

Netowrk: Binance smart chain

Token Type: BEP20

Owner:
0x61B685CF876c8fad49E391Db5b7EC26cBD88fd5E
(at time of audit)

Deployer:
0x61B685CF876c8fad49E391Db5b7EC26cBD88fd5E

Token Address :
0xAFF8a44901e2f342C5cF1833fCD884950a730662

Token Information

Fees:

Buy Fees: up to 6%

Sell Fees: up to 6%

Transfer Fees: 0%

Fees Privilige: Owner

Ownership : Owner

Minting: No

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Priviliges: Including in fees, excluding from fees, 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**	1
◆ **Medium-Risk**	0
◆ **Low-Risk**	0
◆ **Gas Optimization / Suggestions**	0

INHERITANCE TREE
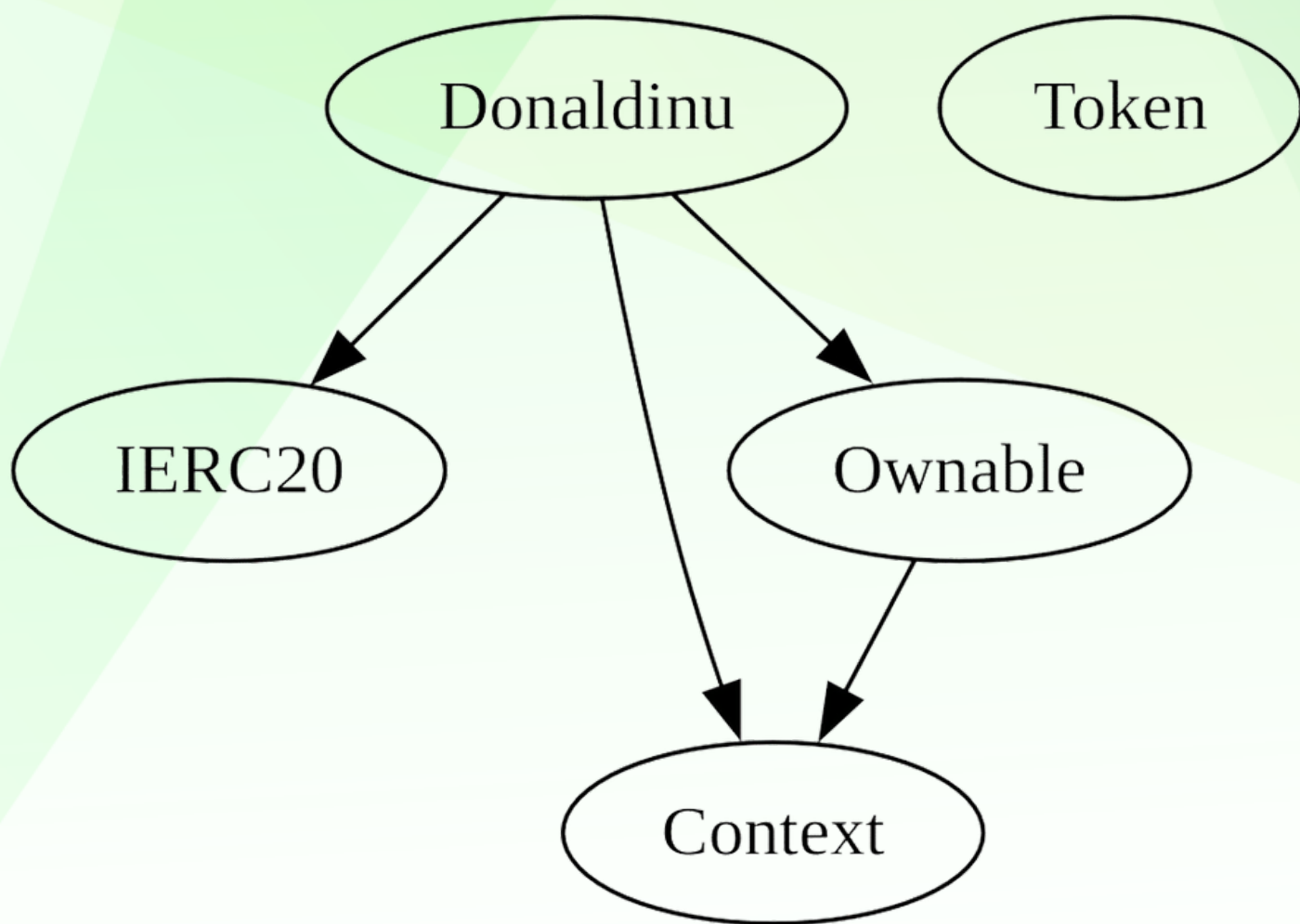
POINTS TO NOTE

- Owner is not able to modify fees (6% for buy/sell and 0% for transfers)
- 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**
IERC20	Interface				
└	totalSupply	External ❗	|NO ❗	
└	balanceOf	External ❗	|NO ❗	
└	transfer	External ❗	🔴 |NO ❗	
└	allowance	External ❗	|NO ❗	
└	approve	External ❗	🔴 |NO ❗	
└	transferFrom	External ❗	🔴 |NO ❗	
Token	Interface				
└	transferFrom	External ❗	🔴 |NO ❗	
└	transfer	External ❗	🔴 |NO ❗	
IUniswapV2Factory	Interface				
└	createPair	External ❗	🔴 |NO ❗	
IUniswapV2Router02	Interface				
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🔴 |NO ❗	
└	factory	External ❗	|NO ❗	
└	WETH	External ❗	|NO ❗	
└	addLiquidityETH	External ❗	💵 |NO ❗	
Context	Implementation				
└	_msgSender	Internal 🔒		
SafeMath	Library				
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	div	Internal 🔒		
Ownable	Implementation	Context			
└	<Constructor>	Public ❗	🔴 |NO ❗	
└	owner	Public ❗	|NO ❗	
└	renounceOwnership	Public ❗	🔴	onlyOwner
└	transferOwnership	Public ❗	🔴	onlyOwner
Donaldinu	Implementation	Context, IERC20, Ownable			
└	<Constructor>	Public ❗	🔴 |NO ❗	

CONTRACT ASSESMENT

└	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❗
└	tokenFromReflection	Private 🔒		
└	_approve	Private 🔒	🔴	
└	_transfer	Private 🔒	🔴	
└	swapTokensForEth	Private 🔒	🔴	lockTheSwap
└	sendETHToFee	Private 🔒	🔴	
└	_tokenTransfer	Private 🔒	🔴	
└	rescueForeignTokens	Public ❗	🔴	onlyDev
└	setNewDevAddress	Public ❗	🔴	onlyDev
└	setNewMarketingAddress	Public ❗	🔴	onlyDev
└	_transferStandard	Private 🔒	🔴	
└	_takeTeam	Private 🔒	🔴	
└	_reflectFee	Private 🔒	🔴	
└	<Receive Ether>	External ❗	💵	NO❗
└	_getValues	Private 🔒		
└	_getTValues	Private 🔒		
└	_getRValues	Private 🔒		
└	_getRate	Private 🔒		
└	_getCurrentSupply	Private 🔒		
└	manualswap	External ❗	🔴	NO❗
└	manualsend	External ❗	🔴	NO❗
└	setFee	Public ❗	🔴	onlyDev
└	tradingStatus	Public ❗	🔴	onlyOwner
└	toggleSwap	Public ❗	🔴	onlyDev
└	excludeMultipleAccountsFromFees	Public ❗	🔴	onlyOwner

Symbol	Meaning
🔴	Function can modify state
💵	Function is payable

STATIC ANALYSIS

```
Reentrancy in Donaldinu.transferFrom(address,address,uint256) (contracts/Token.sol#281-296):
        External calls:
        - _transfer(sender,recipient,amount) (contracts/Token.sol#286)
                - _developmentAddress.transfer(amount.div(2)) (contracts/Token.sol#380)
                - _marketingAddress.transfer(amount.div(2)) (contracts/Token.sol#381)
        State variables written after the call(s):
        - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (contracts/Token.sol#287-294)
                - _allowances[owner][spender] = amount (contracts/Token.sol#312)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (contracts/Token.sol#313)
                - approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (contracts/Token.sol#287-294)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4

Variable Donaldinu._transferStandard(address,address,uint256).rTransferAmount (contracts/Token.sol#429) is too similar to Donaldinu._transferStandard(address,address,uint256).tTra
nsferAmount (contracts/Token.sol#431)
Variable Donaldinu._getValues(uint256).rTransferAmount (contracts/Token.sol#468) is too similar to Donaldinu._transferStandard(address,address,uint256).tTransferAmount (contracts/
Token.sol#431)
Variable Donaldinu._getValues(uint256).rTransferAmount (contracts/Token.sol#468) is too similar to Donaldinu._getValues(uint256).tTransferAmount (contracts/Token.sol#462)
Variable Donaldinu._getValues(uint256).rTransferAmount (contracts/Token.sol#468) is too similar to Donaldinu._getTValues(uint256,uint256,uint256).tTransferAmount (contracts/Token.
sol#484)
Variable Donaldinu._transferStandard(address,address,uint256).rTransferAmount (contracts/Token.sol#429) is too similar to Donaldinu._getValues(uint256).tTransferAmount (contracts/
Token.sol#462)
Variable Donaldinu._transferStandard(address,address,uint256).rTransferAmount (contracts/Token.sol#429) is too similar to Donaldinu._getTValues(uint256,uint256,uint256).tTransferA
mount (contracts/Token.sol#484)
Variable Donaldinu._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (contracts/Token.sol#497) is too similar to Donaldinu._transferStandard(address,address,uint256).tT
ransferAmount (contracts/Token.sol#431)
Variable Donaldinu._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (contracts/Token.sol#497) is too similar to Donaldinu._getValues(uint256).tTransferAmount (contract
s/Token.sol#462)
Variable Donaldinu._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (contracts/Token.sol#497) is too similar to Donaldinu._getTValues(uint256,uint256,uint256).tTransfe
rAmount (contracts/Token.sol#484)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

Donaldinu.slitherConstructorConstantVariables() (contracts/Token.sol#165-565) uses literals with too many digits:
        - _tTotal = 10000000000 * 10 ** 9 (contracts/Token.sol#173)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

Donaldinu._tOwned (contracts/Token.sol#168) is never used in Donaldinu (contracts/Token.sol#165-565)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

Donaldinu.uniswapV2Pair (contracts/Token.sol#196) should be immutable
Donaldinu.uniswapV2Router (contracts/Token.sol#195) 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 issues found

FUNCTIONAL TESTING

1- Adding liquidity (passed):
https://testnet.bscscan.com/tx/0x5b06ec53231d790c37e223a1bd
b4b9762bfe165b7cb428f919f86e7af35c9817

2- Buying when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0xf864bb25cfc1727616be20aacff9
dabdd935af0d043ba6819177ac01a036ff04

3- Selling when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0x38752fa82140760421a899d3d6
9f681c3639fde8e984bb25c06f7262121c5d09

4- Transferring when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0x5d61bd473ddca281f03bc7b433
e002f92b5db38f73bfa877d70b80e04e760268

5- Buying when not excluded from fees (6% tax) (passed):
https://testnet.bscscan.com/tx/0x3a868f0759e69611d002f396ecc
f0b45d8e36f8a0655bbf157a6e2a6307b1e0d

6- Selling when not excluded from fees (6% tax) (passed):
https://testnet.bscscan.com/tx/0xc66f33cfd393662c528e2f2c93
2a24d7f62fe569831fb1e97cd203609e21fb5b

7- Transferring when not excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0x4216a1a63703f565c13c8682ba7
56ded0e59c5a73e50112ce93ae2defe9f0f80

FUNCTIONAL TESTING

8- Internal swap **(passed):**
Marketing and development wallets received BNB
https://testnet.bscscan.com/address/0x1a41c7B57B5265eF5d34f
9180055D633a895bb6F#internaltx
https://testnet.bscscan.com/address/0x61B685CF876c8fad49E39
1Db5b7EC26cBD88fd5E#internaltx

MANUAL TESTING

Centralization - Owner must enable trading

Severity: **High**

Function: tradingStatus

Lines: 549

Status: Not Resolved

Overview:

The owner must activate trading for investors to buy, sell, or transfer tokens. If trading remains disabled, token holders will be unable to trade their tokens.

```
function tradingStatus() public onlyOwner {
    tradingOpen = true;
}
```

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