auditace.tech

AuditAce

FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

Dollar Shiba Inu

DATED : 04 APR 23'

AUDIT SUMMARY

Project name – Ted Inu

Date: 02 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	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 Testnet network:
all tests were done on Bsc Testnet network, each test has its transaction has attached to it.

3- Slither : Static Analysis

Testnet Link: Contract has been tested on binance smart chain testnet which can be found in below link:

https://testnet.bscscan.com/address/0xF030fcDDD000eb539DCB7bD42709a566AdA37432#code

Token Information

Token Name : Dollar Shiba Inu

Token Symbol: DShib

Decimals: 9

Token Supply: 1,000,000,000,000,000,000,000

Token Address:
0x3aB414b56590A5376f5Bd18a57f5680736bE4D98

Checksum:
86f31dd4eaabf2f175342ef2820339554c9a3bdb

Owner:
0x0000000000000000000000000000000000000000
(renounced)

TOKEN OVERVIEW

Fees:

Buy Fees: 8%

Sell Fees: 8%

Transfer Fees: 8%

Fees Privilige: none

Ownership : renounced

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Priviliges: none

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

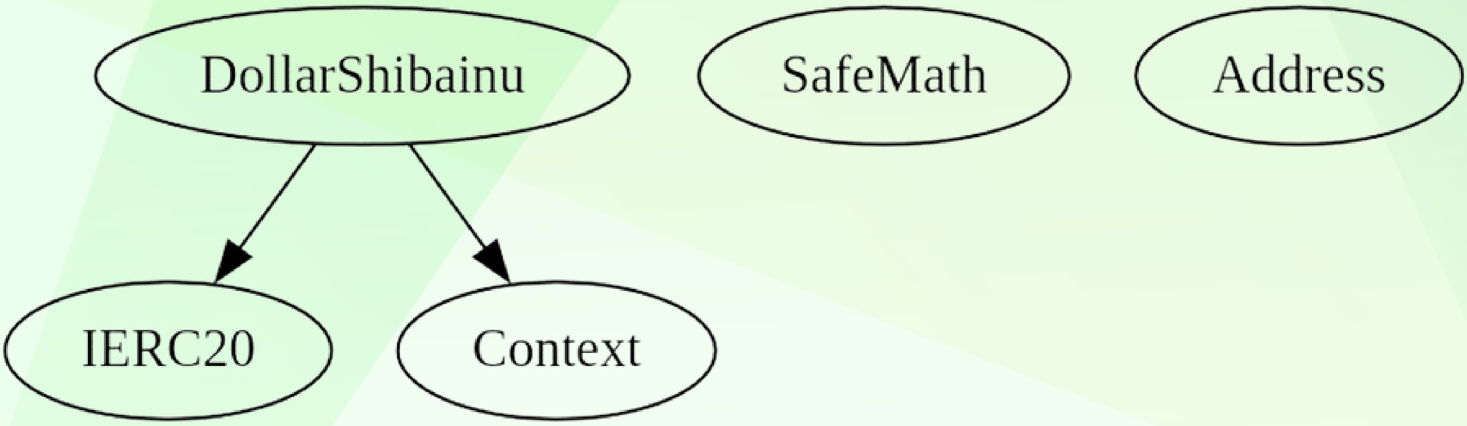
CLASSIFICATION OF RISK

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

- **Ownership is renounced, meaning owner has not control over the contract functions**
- Owner is not able to modify buy/sell/transfer fees (8% for each)
- 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❗ |
||||||
| **SafeMath** | Library | |||
| └ | add | Internal 🔒 | | |
| └ | sub | Internal 🔒 | | |
| └ | mul | Internal 🔒 | | |
| └ | div | Internal 🔒 | | |
| └ | sub | Internal 🔒 | | |
| └ | div | Internal 🔒 | | |
||||||
| **Context** | Implementation | |||
| └ | _msgSender | Internal 🔒 | | |
| └ | _msgData | Internal 🔒 | | |
||||||
| **Address** | Library | |||
| └ | isContract | Internal 🔒 | | |
| └ | sendValue | Internal 🔒 | 🛑 | |
| └ | functionCall | Internal 🔒 | 🛑 | |
| └ | functionCall | Internal 🔒 | 🛑 | |
| └ | functionCallWithValue | Internal 🔒 | 🛑 | |
| └ | functionCallWithValue | Internal 🔒 | 🛑 | |
| └ | functionStaticCall | Internal 🔒 | | |
| └ | functionStaticCall | Internal 🔒 | | |
| └ | functionDelegateCall | Internal 🔒 | 🛑 | |
| └ | functionDelegateCall | Internal 🔒 | 🛑 | |
| └ | _verifyCallResult | Private 🔐 | | |
||||||
| **IUniswapV2Factory** | Interface | |||
| └ | feeTo | External ❗ | |NO❗ |
| └ | feeToSetter | External ❗ | |NO❗ |
| └ | getPair | External ❗ | |NO❗ |
| └ | allPairs | External ❗ | |NO❗ |
```

CONTRACT ASSESMENT

```
| └ | 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❗ |
| └ | 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❗ |
```

CONTRACT ASSESMENT

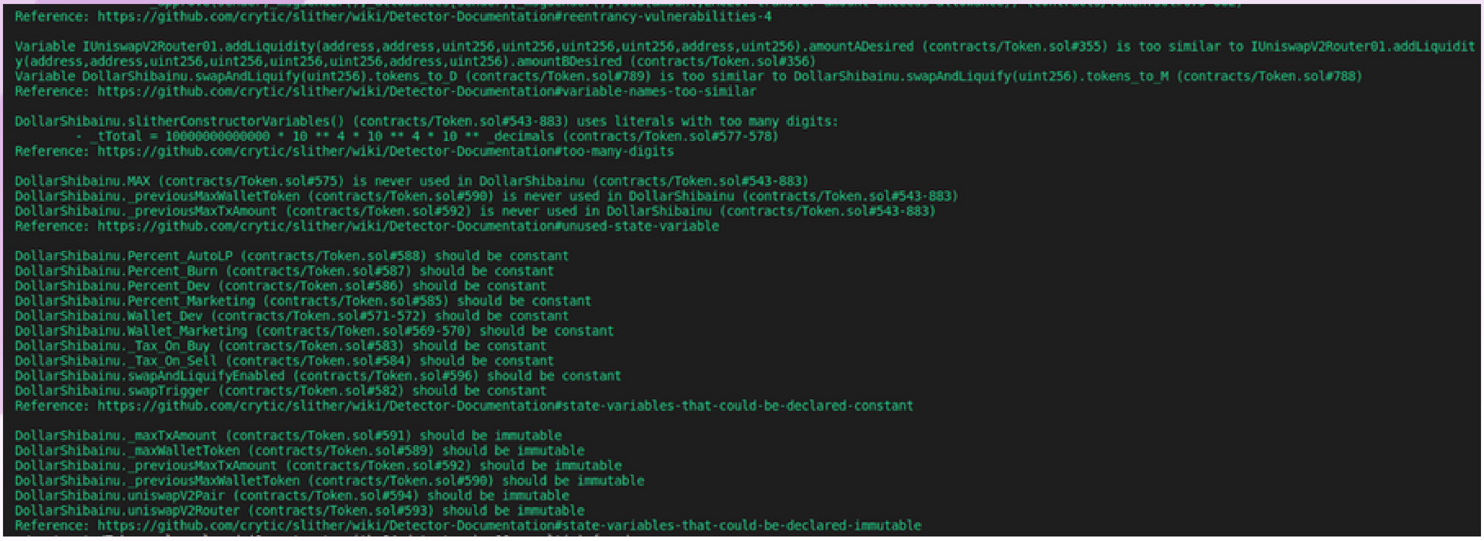
```
| └ | removeLiquidityETHWithPermit | External ❗ | 🛑  |NO❗ |
| └ | swapExactTokensForTokens | External ❗ | 🛑  |NO❗ |
| └ | swapTokensForExactTokens | External ❗ | 🛑  |NO❗ |
| └ | swapExactETHForTokens | External ❗ |  💵 |NO❗ |
| └ | swapTokensForExactETH | External ❗ | 🛑  |NO❗ |
| └ | swapExactTokensForETH | 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❗ |
||||||
| **DollarShibainu** | Implementation | Context, IERC20 |||
| └ | owner | Public ❗ |   |NO❗ |
| └ | renounceOwnership | Public ❗ | 🛑  |NO❗ |
| └ | <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❗ |
| └ | <Receive Ether> | External ❗ |  💵 |NO❗ |
| └ | _getCurrentSupply | Private 🔐 |   | |
| └ | _approve | Private 🔐 | 🛑  | |
| └ | _transfer | Private 🔐 | 🛑  | |
| └ | sendToWallet | Private 🔐 | 🛑  | |
| └ | swapAndLiquify | Private 🔐 | 🛑  | lockTheSwap |
```

└	swapTokensForBNB	Private 🔐	🛑	
└	addLiquidity	Private 🔐	🛑	
└	remove_Random_Tokens	Public ❗️	🛑	NO❗️
└	_tokenTransfer	Private 🔐	🛑	

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS



```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4

Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (contracts/Token.sol#355) is too similar to IUniswapV2Router01.addLiquidit
y(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/Token.sol#356)
Variable DollarShibainu.swapAndLiquify(uint256).tokens_to_D (contracts/Token.sol#789) is too similar to DollarShibainu.swapAndLiquify(uint256).tokens_to_M (contracts/Token.sol#788)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

DollarShibainu.slitherConstructorVariables() (contracts/Token.sol#543-883) uses literals with too many digits:
        - _tTotal = 10000000000000 * 10 ** 4 * 10 ** 4 * 10 ** _decimals (contracts/Token.sol#577-578)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

DollarShibainu.MAX (contracts/Token.sol#575) is never used in DollarShibainu (contracts/Token.sol#543-883)
DollarShibainu._previousMaxWalletToken (contracts/Token.sol#590) is never used in DollarShibainu (contracts/Token.sol#543-883)
DollarShibainu._previousMaxTxAmount (contracts/Token.sol#592) is never used in DollarShibainu (contracts/Token.sol#543-883)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

DollarShibainu.Percent_AutoLP (contracts/Token.sol#588) should be constant
DollarShibainu.Percent_Burn (contracts/Token.sol#587) should be constant
DollarShibainu.Percent_Dev (contracts/Token.sol#586) should be constant
DollarShibainu.Percent_Marketing (contracts/Token.sol#585) should be constant
DollarShibainu.Wallet_Dev (contracts/Token.sol#571-572) should be constant
DollarShibainu.Wallet_Marketing (contracts/Token.sol#569-570) should be constant
DollarShibainu._Tax_On_Buy (contracts/Token.sol#583) should be constant
DollarShibainu._Tax_On_Sell (contracts/Token.sol#584) should be constant
DollarShibainu.swapAndLiquifyEnabled (contracts/Token.sol#596) should be constant
DollarShibainu.swapTrigger (contracts/Token.sol#582) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

DollarShibainu._maxTxAmount (contracts/Token.sol#591) should be immutable
DollarShibainu._maxWalletToken (contracts/Token.sol#589) should be immutable
DollarShibainu._previousMaxTxAmount (contracts/Token.sol#592) should be immutable
DollarShibainu._previousMaxWalletToken (contracts/Token.sol#590) should be immutable
DollarShibainu.uniswapV2Pair (contracts/Token.sol#594) should be immutable
DollarShibainu.uniswapV2Router (contracts/Token.sol#593) 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

Router (PCS V2):
0xD99D1c33F9fC3444f8101754aBC46c52416550D1

1- Adding Liquidity (Passed):
liquidity added on Pancakeswap V2:

https://testnet.bscscan.com/tx/0x1e8d0c57f074f785d429d1d51f85c2910ea23753bd21d53ff339cce99ad2546c

2- Buying when excluded (0%)(Passed):

https://testnet.bscscan.com/tx/0x4e80f7c73bf18843439c4a06d43602a2fd0ad9944be6c4138a77347d6581bd1c

3- Selling when excluded (0%)(Passed):

https://testnet.bscscan.com/tx/0xe9de80d71697ef9cab5414c997b239729df1e29e203bf66891bb0732c31d4a78

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

https://testnet.bscscan.com/tx/0xb6256d59a93e194899b8307a9fbee5bedfc4e3ebb484c69faeb56a11ba518770

5- Buying when not excluded from fees (8% tax) (passed):

https://testnet.bscscan.com/tx/0x89a840cf9e66dac36ba490ef2abe20e37341d67ba78736885f9b404695860343

FUNCTIONAL TESTING

6- Selling when not excluded from fees (8% tax) (passed):

https://testnet.bscscan.com/tx/0xde8881e63eea20aa0e4a4d1865348cdfcf37448104806e510a951bf7f9b1e922

7- Transferring when not excluded from fees (8% tax) (passed):

https://testnet.bscscan.com/tx/0x088ae891bb23ab3fc805359d9f960ac6ee2f11d1ff48f2ecb011e077de5936ee

MANUAL TESTING

No issues found

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