OVERVIEW

PROJECT SUMMARY

Project	**Doitdoge**
Platform	N/A
Language	Solidity

AUDIT SUMMARY

Date	16-03-2023	
Audit Type	Static Analysis, Manual Review	
Audit Result	**PASSED**	
Auditor	**Jarmo van de Seijp**	https://tinyurl.com/Jvdseijp

RISK SUMMARY

Risk Level	Total	Found	Pending	Resolved	Acknowledgde	Objected
Critical	0	0	0	0	0	0
Major	0	0	0	0	0	0
Medium	1	1	1	0	0	0
Minor	3	3	3	0	0	0
Informative	5	5	5	0	0	0
Discussion	0	0	0	0	0	0

FINDINGS

Code order

Description:

The variables
benefactor;
amount;

Can be placed after

```
if (!_exchangePools.contains(beneficiary)) {
        return;
    }
```

Since the variables are not taken into account in case the function returns

Category	Risk Level	Number of Findings	Status
Optimization	informational	**1**	Pending

Redundant zero-address check

Description:

```
require(
from != address(0),
"DoItDoge:_transfer:FROM_ZERO: Cannot transfer from the zero address." );
```

This piece of code is redundant, since the **msg.sender** will never evaluate to address(0);

Category	Risk Level	Number of Findings	Status
zero-adress	informational	**1**	pending

Centralized privilege

Description:

The owner can use all privileged functions to change major variables that affect the project's dynamics. Having unbounded access to major variables carries inherent risk when the owner wallet gets compromised

Note from the Auditor:
The project's team understands the potential risk of centralized privilege, and takes extra steps to secure the owner's wallet(s).

Category	Risk Level	Number of Findings	Status
centralization	Medium	**1**	Pending

Constable State

SWC-ID: SWC-100

Relationship:
CWE-710: Improper Adherence to Coding Standards

Description:
Constant state variables should be declared constant to save gas.

IERC20 public token;

Should either be set as immutable, or hardcoded as constant.

Category	Risk Level	Number of Findings	Status
Constable State	Informative	**2**	pending

Missing Events

Description:

The ***receive() external payable {}*** Method is a safe way to have the contract accept ETH. However, in DeFi contracts it may be difficult to determine the flow of ETH if it is ' blindly' received. Adding an event will help track the financial flow for users and third party aggregators.

Category	Risk Level	Number of Findings	Status
Missing events	informational	**1**	Pending

Push-Over-Pull

Relationship:
CWE-710: Improper Adherence to Coding Standards

Description:

The transfer of the contract's ownership through the function **transferOwnership()** only has 1 check, which is to ensure that the new owner is not the 0 address. It does not, however, check whether or not the ownership can be accepted by the recipient **newOwner**. In the case of a transfer of ownership to an incorrect address, or a smart contract that is not able to use the privileged functions, ownership of the contract is lost permanently with no way of getting it back. It is therefore advisable to use a pull method as opposed to push, in which case the **newOwner** would have to pro-actively accept ownership upon receiving it.

Category	Risk Level	Number of Findings	Status
Push over Pull	Minor	**3**	pending

AUDIT RESULT

Basic Coding Bugs

1. Constructor Mismatch

o Description: Whether the contract name and its constructor are not identical to each other.

o Result: PASSED

o Severity: Critical

Ownership Takeover

o Description: Whether the set owner function is not protected.

o Result: PASSED

o Severity: Critical

Redundant Fallback Function

o Description: Whether the contract has a redundant fallback function.

o Result: PASSED

o Severity: Critical

Overflows & Underflows

Description: Whether the contract has general overflow or underflow Vulnerabilities

o Result: PASSED

o Severity: Critical

Reentrancy

o Description: Reentrancy is an issue when code can call back into your contract and change state, such as withdrawing ETHs.

o Result: PASSED

o Severity: Critical

MONEY-Giving Bug

o Description: Whether the contract returns funds to an arbitrary address.

o Result: PASSED

o Severity: High

Blackhole

- *Description: Whether the contract locks ETH indefinitely: merely in without out.*
- *Result: PASSED*
- *Severity: High*

Unauthorized Self-Destruct

- *Description: Whether the contract can be killed by any arbitrary address.*
- *Result: PASSED*
- *Severity: Medium*

Revert DoS

- *Description: Whether the contractis vulnerable to DoSattack because of unexpected revert.*
- *Result: PASSED*
- *Severity: Medium*

Unchecked External Call

- *Description: Whether the contract has any external call without checking the return value.*
- *Result: PASSED*
- *Severity: Medium*

Gasless Send

- *Description: Whether the contractis vulnerable to gasless send.*
- *Result: PASSED*
- *Severity: Medium*

Send Instead of Transfer

- *Description: Whether the contract uses send instead of transfer.*
- *Result: PASSED*
- *Severity: Medium*

Costly Loop

o Description: Whether the contract has any costly loop which may lead to Out-Of-Gas exception.

o Result: PASSED

o Severity: Medium

(Unsafe) Use of Untrusted Libraries

o Description: Whether the contract use any suspicious libraries.

o Result: PASSED

o Severity: Medium

(Unsafe) Use of Predictable Variables

o Description: Whether the contract contains any randomness variable, but its value can be predicated.

o Result: PASSED

o Severity: Medium

Transaction Ordering Dependence

o Description: Whether the final state of the contract depends on the order of the transactions.

o Result: PASSED

o Severity: Medium

. Deprecated Uses

o Description: Whether the contract use the deprecated tx.origin to perform the authorization.

o Result: PASSED

o Severity: Medium