



# Smart Contract Audit

FOR  
**DOGBIE**

DATED : 3 august 23'



# MANUAL TESTING

## Centralization – Enabling Trades

**Severity:** High

**function:** go\_live

**Status:** Open

### Overview:

The owner of the contract must enable trades manually for investors, otherwise no one would be able to buy/sell/transfer their tokens (even owner of whitelisted wallets)

```
function enableTrading() external onlyOwner {  
    require(!tradingEnabled, "Trading is already enabled");  
    tradingEnabled = true;  
    startTradingBlock = block.number;  
}
```

### Suggestion

It's suggested to either enable trades prior to presale, or transfer ownership of the contract to a certified pinsksale safu developer to guarantee enabling of trades.



# AUDIT SUMMARY

**Project name -DOGBIE**

**Date:** 3 august, 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/0x54BcCACDC894fecA873E7A2eB2c0b5af5e0e69ae>

---



# Token Information

---

**Token Name :** Dogbie

**Token Symbol:** DOGBIE

**Decimals:** 18

**Token Supply:** 100,000,000

**Token Address:**

0x92a2C76303C8Ccc5B26e7994FD0c8ec9Db666c10

**Checksum:**

eb882efe61e737d7322a6ee0e3e9321fdcd0c81c

**Owner:**

**0xC58FFC041806c4FFE34943C27703A3a7FdA5baEf**  
**(at time of writing the audit)**

**Deployer:**

**0xC58FFC041806c4FFE34943C27703A3a7FdA5baEf**



# TOKEN OVERVIEW

---

## Fees:

Buy Fees: 0%

Sell Fees: 2%

Transfer Fees: 0%

---

**Fees Privilege:** static fees

---

**Ownership:** owned

---

**Minting:** No mint function

---

**Max Tx Amount/ Max Wallet Amount:** no

---

**Blacklist:** No

---

**Other Privileges:** Initial distribution of the tokens enabling trades

---



# 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 is exercised 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



**Gasless Send**



Private modifier



Using block.timestamp



Multiple Sends



Re-entrancy



Using Suicide



Tautology or contradiction



Gas Limit and Loops



Timestamp Dependence



Address hardcoded



Revert/require functions



Exception Disorder



Use of tx.origin



Using inline assembly



Integer overflow/underflow



Divide before multiply



Dangerous strict equalities



Missing Zero Address Validation



Using SHA3



Compiler version not fixed



Using throw



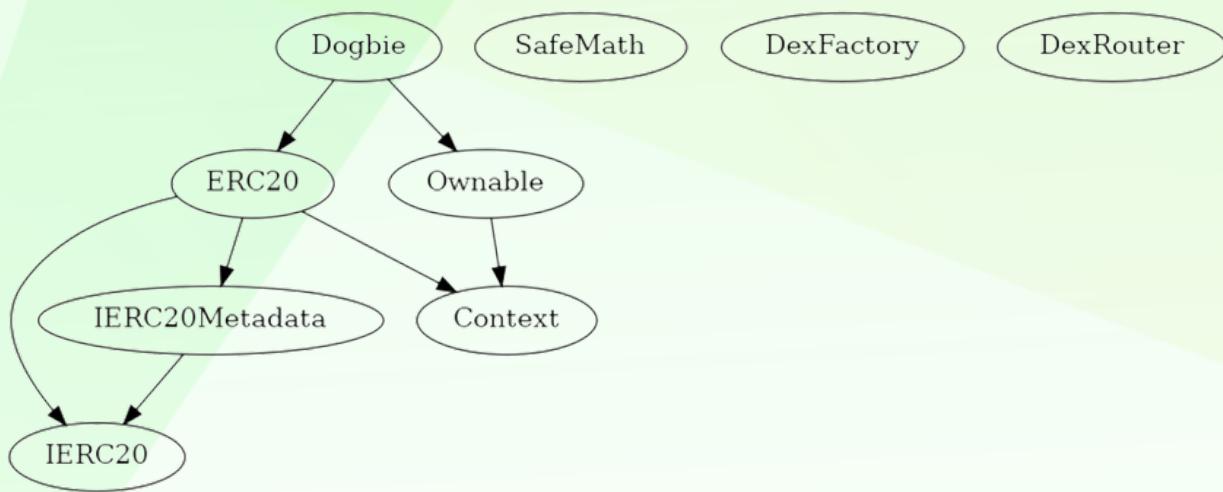
# CLASSIFICATION OF RISK

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 set fee on buy and transfers
- There is 2% static burn tax on sells
- Owner is not able to blacklist an arbitrary address.
- Owner is not able to mint new tokens
- Owner is not able to set max buy/sell/transfer
- **Owner must enable trading for investors**

# CONTRACT ASSESSMENT

---

Contract	Type	Bases			
L   **Function Name**   **Visibility**   **Mutability**   **Modifiers**					
**IERC20**   Interface					
L   totalSupply   External !    NO !					
L   balanceOf   External !    NO !					
L   transfer   External !     NO !					
L   allowance   External !    NO !					
L   approve   External !     NO !					
L   transferFrom   External !     NO !					
**IERC20Metadata**   Interface   IERC20					
L   name   External !    NO !					
L   symbol   External !    NO !					
L   decimals   External !    NO !					
**Context**   Implementation					
L   _msgSender   Internal					
L   _msgData   Internal					
**ERC20**   Implementation   Context, IERC20, IERC20Metadata					
L   <Constructor>   Public !     NO !					
L   name   Public !    NO !					
L   symbol   Public !    NO !					
L   decimals   Public !    NO !					
L   totalSupply   Public !    NO !					
L   balanceOf   Public !    NO !					

# CONTRACT ASSESSMENT

---

```

| L | transfer | Public ! | 🔴 | NO ! | |
| L | allowance | Public ! | | NO ! |
| L | approve | Public ! | 🔴 | NO ! |
| L | transferFrom | Public ! | 🔴 | NO ! |
| L | increaseAllowance | Public ! | 🔴 | NO ! |
| L | decreaseAllowance | Public ! | 🔴 | NO ! |
| L | _transfer | Internal 🔒 | 🔴 |||
| L | _mint | Internal 🔒 | 🔴 |||
| L | _burn | Internal 🔒 | 🔴 |||
| L | _approve | Internal 🔒 | 🔴 |||
| L | _beforeTokenTransfer | Internal 🔒 | 🔴 |||
| L | _afterTokenTransfer | Internal 🔒 | 🔴 |||
|||...
| **Ownable** | Implementation | Context || | |
| L | <Constructor> | Public ! | 🔴 | NO ! |
| L | owner | Public ! | | NO ! |
| L | renounceOwnership | Public ! | 🔴 | onlyOwner |
| L | transferOwnership | Public ! | 🔴 | onlyOwner |
| L | _setOwner | Private 🔒 | 🔴 |||
|||...
| **SafeMath** | Library | || | |
| L | tryAdd | Internal 🔒 | |||
| L | trySub | Internal 🔒 | |||
| L | tryMul | Internal 🔒 | |||
| L | tryDiv | Internal 🔒 | |||
| L | tryMod | Internal 🔒 | |
| L | add | Internal 🔒 | |||
| L | sub | Internal 🔒 | |||
| L | mul | Internal 🔒 | |||
| L | div | Internal 🔒 | |||
| L | mod | Internal 🔒 | |||
| L | sub | Internal 🔒 | |||
| L | div | Internal 🔒 |
| L | mod | Internal 🔒 |
|||...
| **DexFactory** | Interface | ||
| L | createPair | External ! | 🔴 | NO ! |

```

# CONTRACT ASSESSMENT

---

```

||||| |
| **DexRouter** | Interface | ||
| L| factory | External ! | |NO ! |
| L| WETH | External ! | |NO ! |
| L| addLiquidityETH | External ! | 💸 |NO ! |
| L| swapExactTokensForETHSupportingFeeOnTransferTokens | External ! | 💳 |NO ! |
|||||
| **Dogbie** | Implementation | ERC20, Ownable ||
| L| <Constructor> | Public ! | 💳 | ERC20 |
| L| enableTrading | External ! | 💳 | onlyOwner |
| L| setWhitelistStatus | External ! | 💳 | onlyOwner |
| L| checkWhitelist | External ! | |NO ! |
| L| _takeTax | Internal 🔒 | 💳 |
| L| _transfer | Internal 🔒 | 💳 |
| L| withdrawStuckETH | External ! | 💳 | onlyOwner |
| L| withdrawStuckTokens | External ! | 💳 | onlyOwner |
| L| <Receive Ether> | External ! | 💸 |NO ! |

```

### ### Legend

Symbol	Meaning
----- -----	
🔴	Function can modify state
💸	Function is payable



# STATIC ANALYSIS

Context.\_msgData() (contracts/Token.sol#45-47) is never used and should be removed  
ERC20.\_burn(address,uint256) (contracts/Token.sol#188-203) is never used and should be removed  
SafeMath.add(uint256,uint256) (contracts/Token.sol#313-315) is never used and should be removed  
SafeMath.div(uint256,uint256) (contracts/Token.sol#325-327) is never used and should be removed  
SafeMath.div(uint256,uint256,string) (contracts/Token.sol#344-353) is never used and should be removed  
SafeMath.mod(uint256,uint256) (contracts/Token.sol#329-331) is never used and should be removed  
SafeMath.mod(uint256,uint256,string) (contracts/Token.sol#355-364) is never used and should be removed  
SafeMath.mul(uint256,uint256) (contracts/Token.sol#321-323) is never used and should be removed  
SafeMath.sub(uint256,uint256) (contracts/Token.sol#317-319) is never used and should be removed  
SafeMath.sub(uint256,uint256,string) (contracts/Token.sol#333-342) is never used and should be removed  
SafeMath.tryAdd(uint256,uint256) (contracts/Token.sol#272-278) is never used and should be removed  
SafeMath.tryDiv(uint256,uint256) (contracts/Token.sol#299-304) is never used and should be removed  
SafeMath.tryMod(uint256,uint256) (contracts/Token.sol#306-311) is never used and should be removed  
SafeMath.tryMul(uint256,uint256) (contracts/Token.sol#287-297) is never used and should be removed  
SafeMath.trySub(uint256,uint256) (contracts/Token.sol#280-285) 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#6) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16  
solc-0.8.21 is not recommended for deployment  
Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity>

Low level call in Dogbie.withdrawStuckETH() (contracts/Token.sol#498-503):  
- (success) = address(msg.sender).call{value: balance}()  
Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls>

Function DexRouter.WETH() (contracts/Token.sol#377) is not in mixedCase  
Parameter Dogbie.setWhitelistStatus(address,bool).\_wallet (contracts/Token.sol#446) is not in mixedCase  
Parameter Dogbie.setWhitelistStatus(address,bool).\_status (contracts/Token.sol#447) is not in mixedCase  
Parameter Dogbie.checkWhitelist(address).\_wallet (contracts/Token.sol#453) is not in mixedCase  
Parameter Dogbie.withdrawStuckTokens(address).ERC20\_token (contracts/Token.sol#505) is not in mixedCase  
Constant Dogbie.\_totalSupply (contracts/Token.sol#401) is not in UPPER CASE WITH underscores  
Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions>

Dogbie.burnAddress (contracts/Token.sol#418) should be constant  
Dogbie.sellTaxes (contracts/Token.sol#408) should be constant  
Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant>

**Result => A static analysis of contract's source code has been performed using slither,  
No major issues were found in the output**



# FUNCTIONAL TESTING

---

## 1- Adding liquidity (**passed**):

<https://testnet.bscscan.com/tx/0x450782280dfb908bfb20c956965b6320c4147f3bfda3c11013c142e4bfa4c9b0>

## 2- Buying when excluded from fees (0% tax) (**passed**):

<https://testnet.bscscan.com/tx/0xef0f20229235e5e30cabac265495bb5aca5133c6b065eb395fadf3b4b71b9b57>

## 3- Selling when excluded from fees (0% tax) (**passed**):

<https://testnet.bscscan.com/tx/0xde992f33fa45ce3da50755480c07b81cd185e2fc026b6c7faf3e8d79a84bdbd6>

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

<https://testnet.bscscan.com/tx/0x862fdf7286c8418b370062a6794e543e6845aeecc5c32617f92c4d20cc1954f96>

## 5- Buying when not excluded from fees (0% tax) (**passed**):

<https://testnet.bscscan.com/tx/0xe297e23a0edb04856cf0e0dd455483421f01ddb049bf3538a3fc7e14736bad4>

## 6- Selling when not excluded from fees (2% tax) (**passed**):

<https://testnet.bscscan.com/tx/0xd08f9634bc3349d7c1347d35ac11448f6d931d5d465137d64b0a69cb034b2928>

---



# FUNCTIONAL TESTING

---

## 7- Transferring (0% tax) (passed):

<https://testnet.bscscan.com/tx/0xc57fc556e1dae4f34c4764c912fadbe8c57d36665a4a45b2b52f760d726d0ade>



# MANUAL TESTING

## Centralization – Enabling Trades

**Severity:** High

**function:** go\_live

**Status:** Open

### Overview:

The owner of the contract must enable trades manually for investors, otherwise no one would be able to buy/sell/transfer their tokens (even owner of whitelisted wallets)

```
function enableTrading() external onlyOwner {  
    require(!tradingEnabled, "Trading is already enabled");  
    tradingEnabled = true;  
    startTradingBlock = block.number;  
}
```

### Suggestion

It's suggested to either enable trades prior to presale, or transfer ownership of the contract to a certified pinsksale safu developer to guarantee enabling of trades.



# 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 specialize 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://t.me/Audit_Ace)**



**[https://twitter.com/auditace\\_](https://twitter.com/auditace_)**



**<https://github.com/Audit-Ace>**

---