AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

WojakMemeCoin

DATED : 27 May 23'

HIGH RISK

Centralization – Trades must be enabled

Severity: **High**

function: startTrading

Status: Not Resolved

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 startTrading() external onlyOwner {
    require(!tradingEnabled, "Trading already enabled");
    tradingEnabled = true;
}
```

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

AUDIT SUMMARY

Project name – WojakMemeCoin

Date: 27 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	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/0x2Daf26D9a1E4a2CA8C172C90c095ea885aD984CD

Token Information

Token Name : WojakMemeCoin

Token Symbol: WojakCoin

Decimals: 9

Token Supply: 1,000,000,000

Token Address:
0xB6Af22E72Fb7ac5Bb8a4E30189CBc2448a16b454

Checksum:
425447c30e0cd2536a177c87168b3952451df73a

Owner:
0x54deB88004936e1d9A312ba1994B8Bf894B7eb72

Deployer:
0x89F30534B602BE37e32d3576BFebc1099DB6b870

TOKEN OVERVIEW

Fees:

Buy Fees: 0-10%

Sell Fees: 0-10%

Transfer Fees: 0-5%

Fees Privilege: Owner

Ownership: 0x54deB88004936e1d9A312ba1994B8Bf894B7eb72

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Privileges: including in fees

excluding from fees

initial distribution of the tokens

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

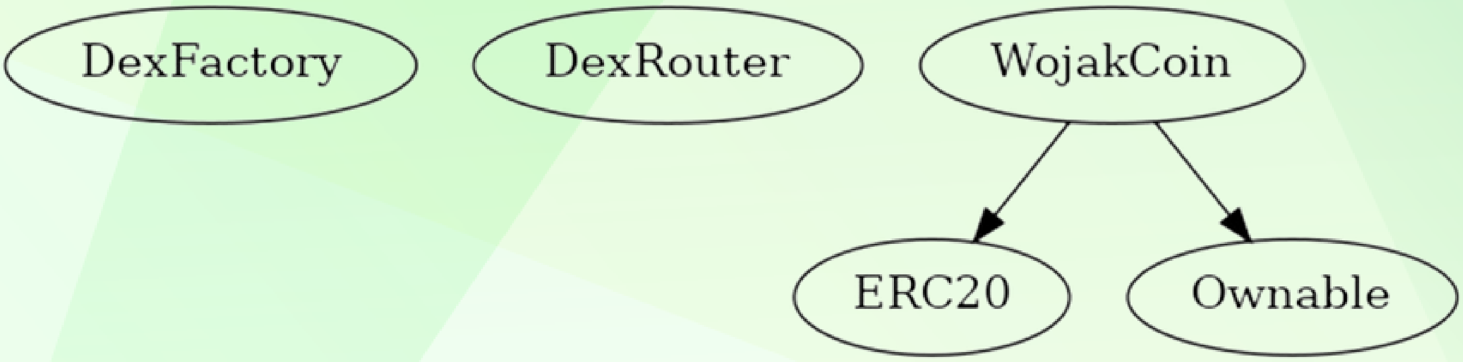
CLASSIFICATION OF RISK

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



CONTRACT ASSESMENT

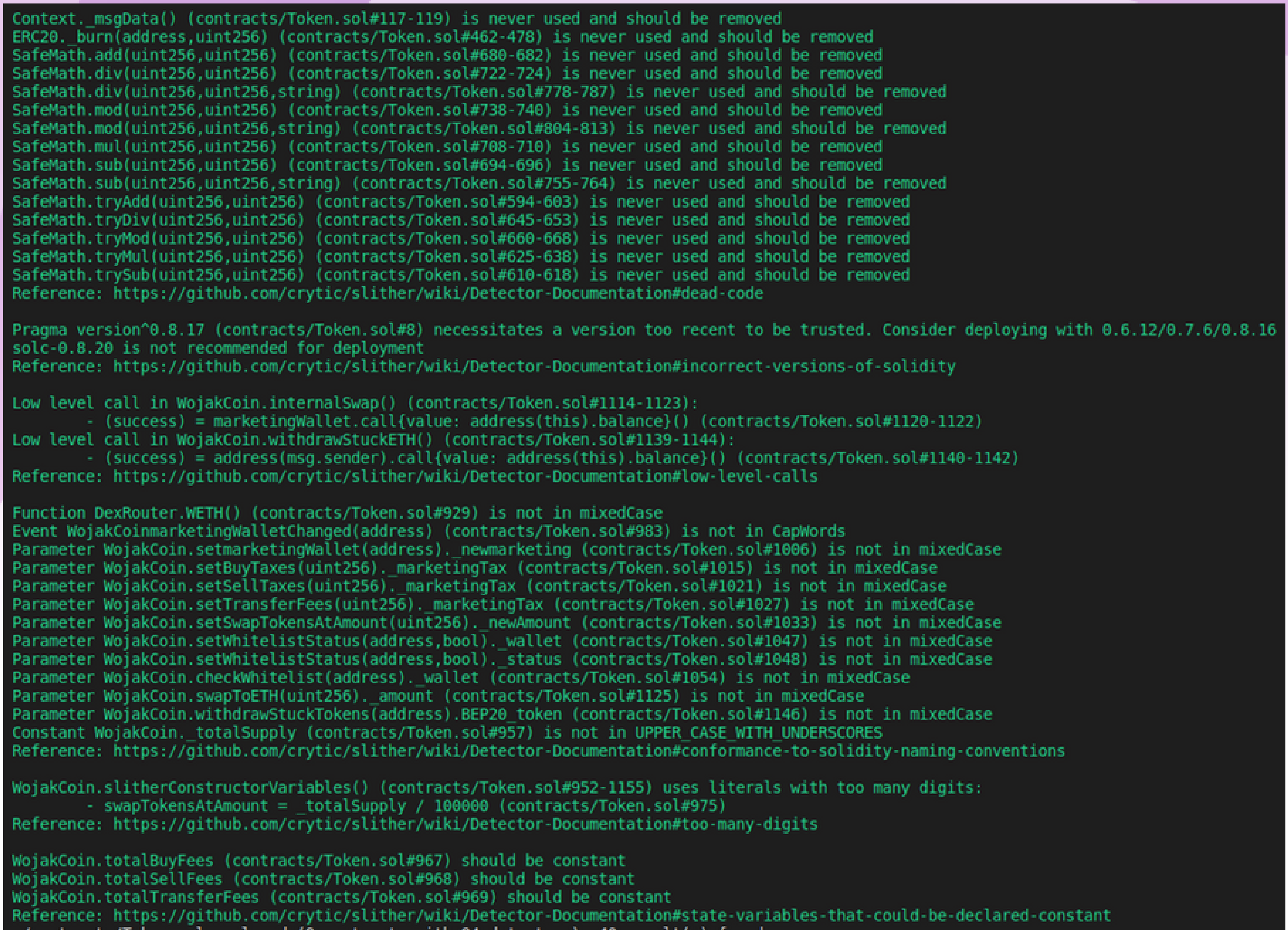
```
| Contract | Type | Bases | | |
|:----------:|:-------------------:|:----------------:|:---------------:|:---------------:|
| └ | **Function Name** | **Visibility** | **Mutability** | **Modifiers** |
||||||
| **DexFactory** | Interface | |||
| └ | createPair | External ❗ | 🛑 |NO ❗ |
||||||
| **DexRouter** | Interface | |||
| └ | factory | External ❗ | |NO ❗ |
| └ | WETH | External ❗ | |NO ❗ |
| └ | addLiquidityETH | External ❗ | 💵 |NO ❗ |
| └ | swapExactTokensForETHSupportingFeeOnTransferTokens | External ❗ | 🛑 |NO ❗ |
||||||
| **WojakCoin** | Implementation | ERC20, Ownable |||
| └ | <Constructor> | Public ❗ | 🛑 | ERC20 |
| └ | setmarketingWallet | External ❗ | 🛑 | onlyOwner |
| └ | setBuyTaxes | External ❗ | 🛑 | onlyOwner |
| └ | setSellTaxes | External ❗ | 🛑 | onlyOwner |
| └ | setTransferFees | External ❗ | 🛑 | onlyOwner |
| └ | setSwapTokensAtAmount | External ❗ | 🛑 | onlyOwner |
| └ | toggleSwapping | External ❗ | 🛑 | onlyOwner |
| └ | setWhitelistStatus | External ❗ | 🛑 | onlyOwner |
| └ | checkWhitelist | External ❗ | |NO ❗ |
| └ | startTrading | External ❗ | 🛑 | onlyOwner |
| └ | _takeTax | Internal 🔒 | 🛑 | |
| └ | _transfer | Internal 🔒 | 🛑 | |
| └ | internalSwap | Internal 🔒 | 🛑 | |
| └ | swapToETH | Internal 🔒 | 🛑 | |
| └ | withdrawStuckETH | External ❗ | 🛑 | onlyOwner |
| └ | withdrawStuckTokens | External ❗ | 🛑 | onlyOwner |
| └ | <Receive Ether> | External ❗ | 💵 |NO ❗ |

| Symbol | Meaning |
|:--------:|-----------|
| 🛑 | Function can modify state |
| 💵 | Function is payable |
```

POINTS TO NOTE

- **Owner is not able to change buy/sell fees over 12% and transfer fee over 5%**
- Owner is not able to blacklist an arbitrary address.
- Owner is not able to disable trades
- Owner is not able to set max buy/sell/transfer/hold amount to 0
- Owner is not able to mint new tokens
- **Owner must enable trades manually**

STATIC ANALYSIS



```
Context._msgData() (contracts/Token.sol#117-119) is never used and should be removed
ERC20._burn(address,uint256) (contracts/Token.sol#462-478) is never used and should be removed
SafeMath.add(uint256,uint256) (contracts/Token.sol#680-682) is never used and should be removed
SafeMath.div(uint256,uint256) (contracts/Token.sol#722-724) is never used and should be removed
SafeMath.div(uint256,uint256,string) (contracts/Token.sol#778-787) is never used and should be removed
SafeMath.mod(uint256,uint256) (contracts/Token.sol#738-740) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (contracts/Token.sol#804-813) is never used and should be removed
SafeMath.mul(uint256,uint256) (contracts/Token.sol#708-710) is never used and should be removed
SafeMath.sub(uint256,uint256) (contracts/Token.sol#694-696) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (contracts/Token.sol#755-764) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (contracts/Token.sol#594-603) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (contracts/Token.sol#645-653) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (contracts/Token.sol#660-668) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (contracts/Token.sol#625-638) is never used and should be removed
SafeMath.trySub(uint256,uint256) (contracts/Token.sol#610-618) 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#8) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.20 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in WojakCoin.internalSwap() (contracts/Token.sol#1114-1123):
        - (success) = marketingWallet.call{value: address(this).balance}() (contracts/Token.sol#1120-1122)
Low level call in WojakCoin.withdrawStuckETH() (contracts/Token.sol#1139-1144):
        - (success) = address(msg.sender).call{value: address(this).balance}() (contracts/Token.sol#1140-1142)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function DexRouter.WETH() (contracts/Token.sol#929) is not in mixedCase
Event WojakCoinmarketingWalletChanged(address) (contracts/Token.sol#983) is not in CapWords
Parameter WojakCoin.setmarketingWallet(address)._newmarketing (contracts/Token.sol#1006) is not in mixedCase
Parameter WojakCoin.setBuyTaxes(uint256)._marketingTax (contracts/Token.sol#1015) is not in mixedCase
Parameter WojakCoin.setSellTaxes(uint256)._marketingTax (contracts/Token.sol#1021) is not in mixedCase
Parameter WojakCoin.setTransferFees(uint256)._marketingTax (contracts/Token.sol#1027) is not in mixedCase
Parameter WojakCoin.setSwapTokensAtAmount(uint256)._newAmount (contracts/Token.sol#1033) is not in mixedCase
Parameter WojakCoin.setWhitelistStatus(address,bool)._wallet (contracts/Token.sol#1047) is not in mixedCase
Parameter WojakCoin.setWhitelistStatus(address,bool)._status (contracts/Token.sol#1048) is not in mixedCase
Parameter WojakCoin.checkWhitelist(address)._wallet (contracts/Token.sol#1054) is not in mixedCase
Parameter WojakCoin.swapToETH(uint256)._amount (contracts/Token.sol#1125) is not in mixedCase
Parameter WojakCoin.withdrawStuckTokens(address).BEP20_token (contracts/Token.sol#1146) is not in mixedCase
Constant WojakCoin._totalSupply (contracts/Token.sol#957) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

WojakCoin.slitherConstructorVariables() (contracts/Token.sol#952-1155) uses literals with too many digits:
        - swapTokensAtAmount = _totalSupply / 100000 (contracts/Token.sol#975)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

WojakCoin.totalBuyFees (contracts/Token.sol#967) should be constant
WojakCoin.totalSellFees (contracts/Token.sol#968) should be constant
WojakCoin.totalTransferFees (contracts/Token.sol#969) 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/0x8fec88e4187cd83d44403b296a90698211ae06a2bc8ebefab94555cd106c6e00

2- Buying when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0xdb5f6fd0df349ec84e3507532a9db48c460e2f106d48c7d228db2c3370f55aeb

3- Selling when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0xafaafeba79de2f9b24694bebe860b5fba61051fa45a64f05fb575ae0af8560df

4- Transferring when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0x295119e035e70915132669d9fc7139d6c330fff5f37dd1938944bd94b3ab0eb1

5- Buying when not excluded from fees (0-12% tax) (passed):
https://testnet.bscscan.com/tx/0xe647effac153c5f3f429673ce5453d8cb3f50788a0cc6b8c2c0bbe7b3c9dcd1a

6- Selling when not excluded from fees (0-12% tax) (passed):
https://testnet.bscscan.com/tx/0x66a1774f0c138990fefeb716d934f3d932dfe5193007ed5f32559de10b01b035

FUNCTIONAL TESTING

7- Transferring when not excluded from fees (0-5% tax) (passed):
https://testnet.bscscan.com/tx/0x41d1b584a089a9dfd1f5db57c532f3c70739a31634ad3e77ef2df06f70446ab9

8- Internal swap (marketing bnb) (passed):
https://testnet.bscscan.com/address/0xa480701222ba660e888cacc62f53259c887cd824#internaltx

MANUAL TESTING

Centralization – Trades must be enabled

Severity: **High**

function: startTrading

Status: Not Resolved

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 startTrading() external onlyOwner {
    require(!tradingEnabled, "Trading already enabled");
    tradingEnabled = true;
}
```

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

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