auditace.tech

AuditAce

FROM INCEPTION TO SUCCESS

# Smart Contract Audit

FOR

# Kings Of Wojak

DATED : 8 September 23'

# MANUAL TESTING

## Centralization – Enabling Trades

**Severity: High**
**function: openTrading**
**Status: Open**

**Overview:**
The openTrading function permits only the contract owner to activate trading capabilities. Until this function is executed, no investors can buy, sell, or transfer their tokens. This places a high degree of control and centralization in the hands of the contract owner.

```
function openTrading() external onlyOwner {
    require(presaleActive == 3, "LP Not Ready");
    presaleActive = 1;
}
```

**Suggestion**
To reduce centralization and potential manipulation, consider one of the following approaches:
1.Automatically enable trading after a specified condition, such as the completion of a presale, is met.
2.If manual activation is still desired, consider transferring the ownership of the contract to a trustworthy, third-party entity like a certified "PinkSale Safu" developer. This can provide investors with more confidence in the eventual activation of trading capabilities, mitigating concerns of potential bad faith actions by the original owner

# MANUAL TESTING

## Centralization – LP tokens sent to owner

**Severity: High**
**function: openTrading**
**Status: Open**
**Overview:**
**After end of the presale, all LP tokens will be sent to owner**

```
        dexRouter.addLiquidityETH{value: address(this).balance}(
            address(this),
            bal,
            0,
            0,
            owner(),
            block.timestamp
        )
```

**Suggestion**
**Its suggested to lock or burn LP tokens.**

# MANUAL TESTING

## Centralization – Blacklist

**Severity: High**
**function: transferProtection**
**Status: Open**
**Overview:**
**Owner is able to blacklist an arbitrary wallet. Blacklisted wallets wont be able to sell or transfer their tokens**

```
function transferProtection(
address[] calldata _wallets,
uint256 _enabled
  ) external onlyOwner {
for (uint256 i = 0; i < _wallets.length; i++) {
walletProtection[_wallets[i]] = _enabled;
}
  }
```

**Suggestion**
**Implement a more decentralized and automated method for blacklisting bad actors (such as using dead blocks or using maximum wallet/buy/sell/transfer limit)**

# MANUAL TESTING

## Logical – Stuck ETH and tokens

**Severity: High**

**function: transferProtection**

**Status: Open**

**Overview:**

**If presale doesn't reach hardcap, there is no way to withdraw deposited ETH from the contract**

**Suggestion**

**Create a function for allowing participates to withdraw their deposited ETH if presale faild.**

# AUDIT SUMMARY

**Project name** – Kings Of Wojak

**Date**: 8 September 2023

**Scope of Audit-** Audit Ace was consulted to conduct the smart contract audit of the solidity source codes.

**Audit Status: FAILED**

## Issues Found

| Status | Critical | High | Medium | Low | Suggestion |
|---|---|---|---|---|---|
| Open | 0 | 4 | 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/0xc89361d77fa219e42cadd6795c8199ecdfdd74a3

# Token Information

**Token Address :**
0x2c14ff70774b7B5d7732cdf13e47a416996B0e6B

**Name:** Kings Of Wojak

**Symbol:** KJAK

**Decimals:** 9

**Network:** Ethereum

**Token Type:** ERC20

**Owner:** 0x2c14ff70774b7B5d7732cdf13e47a416996B0e6B

**Deployer:** 0x7FA05f2c10c21B0f14e47446eBE41bc2CAB6d8eD

**Token Supply:** 69,000,000

**Checksum:**
3aa85371cb9853106409d78434d3d28f551c2fad

**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/0xc89361d77fa219e42cadd6795c8199ecdfdd74a3

# TOKEN OVERVIEW

**buy fee:** 0%

**Sell fee:** 0%

**transfer fee:** 0%

**Fee Privilege:** Owner

**Ownership:** Owned

**Minting:** None

**Max Tx:** None

**Blacklist:** Yes

**Other Privileges:**

- Initial distribution of the tokens
- Enabling trades
- Blacklisting

# 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.

auditace.tech

# 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

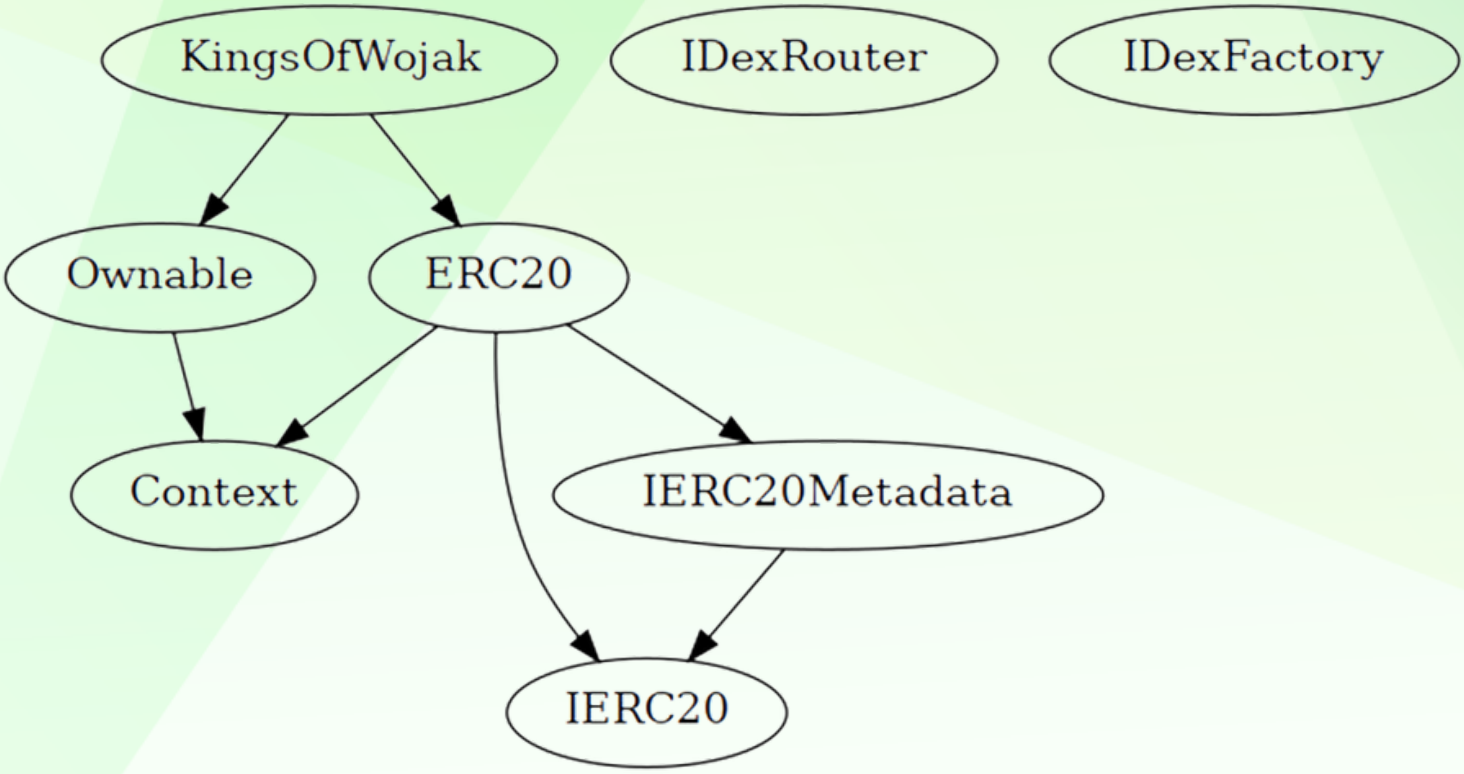
# CLASSIFICATION OF RISK

| 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 | 4 |
| Medium-Risk | 0 |
| Low-Risk | 0 |
| Gas Optimization / Suggestions | 0 |

# INHERITANCE TREE



# POINTS TO NOTE

- Owner is not able to set fee on buy/sell/transfer
- **Owner is able to blacklist an arbitrary wallet (wallet protection)**
- Owner is not able to disable trades
- Owner is not able to mint new tokens
- Owner is not able to set maximum wallet and maximum buy/sell/transfer limits
- **Owner must enable trades manually**

# STATIC ANALYSIS

```
INFO:Detectors:
KingsOfWojak.launch() (contracts/Token.sol#462-495) ignores return value by dexRouter.addLiquidityETH{value: address(this).balance}(address(this),tokensForLP,0,0,owner(),block.timestamp) (con
tracts/Token.sol#481-494)
KingsOfWojak.manualLaunch() (contracts/Token.sol#497-520) ignores return value by dexRouter.addLiquidityETH{value: address(this).balance}(address(this),bal,0,0,owner(),block.timestamp) (contr
acts/Token.sol#502-519)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
KingsOfWojak.constructor().totalSupply (contracts/Token.sol#400) shadows:
        - ERC20.totalSupply() (contracts/Token.sol#149-151) (function)
        - IERC20.totalSupply() (contracts/Token.sol#24) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
Reentrancy in KingsOfWojak.manualLaunch() (contracts/Token.sol#497-520):
        External calls:
        - dexRouter.addLiquidityETH{value: address(this).balance}(address(this),bal,0,0,owner(),block.timestamp) (contracts/Token.sol#502-519)
        - (success,None) = address(owner()).call{value: address(this).balance}() (contracts/Token.sol#515-517)
        Event emitted after the call(s):
        - Transfer(sender,recipient,amount) (contracts/Token.sol#248)
                - super._transfer(address(this),owner(),balanceOf(address(this))) (contracts/Token.sol#518)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
Context._msgData() (contracts/Token.sol#14-17) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/Token.sol#7) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in KingsOfWojak.launch() (contracts/Token.sol#462-495):
        - (success,None) = address(owner()).call{value: address(this).balance / 2}() (contracts/Token.sol#477-479)
Low level call in KingsOfWojak.manualLaunch() (contracts/Token.sol#497-520):
        - (success,None) = address(owner()).call{value: address(this).balance}() (contracts/Token.sol#515-517)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IDexRouter.WETH() (contracts/Token.sol#311) is not in mixedCase
Parameter KingsOfWojak.transferProtection(address[],uint256)._wallets (contracts/Token.sol#548) is not in mixedCase
Parameter KingsOfWojak.transferProtection(address[],uint256)._enabled (contracts/Token.sol#549) is not in mixedCase
Constant KingsOfWojak.routerAddress (contracts/Token.sol#376) is not in UPPER_CASE_WITH_UNDERSCORES
Constant KingsOfWojak.dexRouter (contracts/Token.sol#377) is not in UPPER_CASE_WITH_UNDERSCORES
Constant KingsOfWojak._decimals (contracts/Token.sol#381) is not in UPPER_CASE_WITH_UNDERSCORES
Constant KingsOfWojak._decimalFactor (contracts/Token.sol#382) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (contracts/Token.sol#15)" inContext (contracts/Token.sol#9-18)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Slither:./contracts/Token.sol analyzed (8 contracts with 88 detectors), 20 result(s) found
```

**Result => A static analysis of contract's source code has been performed using slither,**

**No major issues were found in the output**

# CONTRACT ASSESMENT

| Contract| Type |Bases | | |
|:----------:|:----------:|:----------:|:----------:|:----------:|
| └| **Function Name** |**Visibility** | **Mutability** |**Modifiers** |
||||||
| **Context** | Implementation | ||
| └| _msgSender | Internal 🔒 | | |
| └| _msgData | Internal 🔒 | | |
||||||
| **IERC20** | Interface | ||
| └| totalSupply | External ❗ | |NO❗ |
| └| balanceOf | External ❗ | |NO❗ |
| └| transfer | External ❗ | 🛑 |NO❗ |
| └| allowance | External ❗ | |NO❗ |
| └| approve | External ❗ | 🛑 |NO❗ |
| └| transferFrom | External ❗ | 🛑|NO❗ |
||||||
| **IERC20Metadata** | Interface | IERC20 ||
| └| name | External ❗ | |NO❗ |
| └| symbol | External ❗ | |NO❗ |
| └| decimals | External ❗ | |NO❗ |
||||||
| **ERC20** | Implementation | Context, IERC20, IERC20Metadata ||
| └| <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❗ |
| └| _transfer | Internal 🔒 | 🛑| |
| └| _approve | Internal 🔒 | 🛑| |
| └| _initialTransfer | Internal 🔒 | 🛑 | |

# CONTRACT ASSESMENT

||||||
| **Ownable** | Implementation | Context |||
| └| <Constructor> | Public ❗ | 🛑|NO❗ |
| └| owner | Public ❗ | |NO❗ |
| └| renounceOwnership | Public ❗ | 🛑| onlyOwner |
| └| transferOwnership | Public ❗ | 🛑| onlyOwner |
||||||
| **IDexRouter** | Interface | |||
| └| factory | External ❗ | |NO❗ |
| └| WETH | External ❗ | |NO❗ |
| └| swapExactTokensForETHSupportingFeeOnTransferTokens | External ❗ | 🛑 |NO❗ |
| └| swapExactTokensForTokensSupportingFeeOnTransferTokens | External ❗ | 🛑 |NO❗ |
| └| swapExactETHForTokensSupportingFeeOnTransferTokens | External ❗ | 💵 |NO❗ |
| └| swapExactETHForTokens | External ❗ | 💵 |NO❗ |
| └| swapETHForExactTokens | External ❗ | 💵 |NO❗ |
| └| addLiquidityETH | External ❗ | 💵 |NO❗ |
| └| getAmountsOut | External ❗ | |NO❗ |
||||||
| **IDexFactory** | Interface | |||
| └| createPair | External ❗ | 🛑|NO❗ |
||||||
| **KingsOfWojak** | Implementation | ERC20, Ownable |||
| └| <Constructor> | Public ❗ | 🛑| ERC20 |
| └| <Receive Ether> | External ❗ | 💵 |NO❗ |
| └| joinPresale | External ❗ | 💵 |NO❗ |
| └| _presale | Internal 🔒 | 🛑| |
| └| decimals | Public ❗ | |NO❗ |
| └| _transfer | Internal 🔒 | 🛑| |
| └| launch | Public ❗ | 🛑 |NO❗ |
| └| manualLaunch | External ❗ | 🛑 | onlyOwner |
| └| openTrading | External ❗ | 🛑 | onlyOwner |
| └| extractExcessTokens | External ❗ | 🛑| onlyOwner |
| └| airdrop | External ❗ | 🛑 | onlyOwner |
| └| transferProtection | External ❗ | 🛑| onlyOwner |
| └| _beforeTokenTransfer | Internal 🔒 || |

# CONTRACT ASSESMENT

### Legend

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

# FUNCTIONAL TESTING

1- Adding liquidity (after end of presale) (passed):
https://testnet.bscscan.com/tx/0x675cf4dbb62a436ac06ad4ba2e19378240fd5bcfe8ee39afbced377e271ac908

2- Buying (0% tax) (passed):
https://testnet.bscscan.com/tx/0xaae8241412eac8e4f4c1583f9a1dbc7767a758e0160af21d0671da38c8e1b4a2

3- Selling (0% tax) (passed):
https://testnet.bscscan.com/tx/0x673f9dd87c3192a2cba9e2ed3cba0e67e432109af9dd2f9e11aced407f6c2a27

4- Transferring (0% tax) (passed):
https://testnet.bscscan.com/tx/0xd56d0d14154b5d59cf1329021744ecb17cae07a1c1a337e423056d6d1ee76669

5- Participating in presale (10 times) (passed):
https://testnet.bscscan.com/tx/0xd5f4d12716ff6e01ba2d3b44cc40b8aeef63c8048e151b891dc09efebe5d24c8
https://testnet.bscscan.com/tx/0x91886fa725c8b625824a763fef7315cb4ac5f8a0eb1699e348f14845822d460a
https://testnet.bscscan.com/tx/0x8c30ec8838758b123df111324dbebd5bfaab0660ba74a4c52e0f4d3df1436265

# MANUAL TESTING

## Centralization – Enabling Trades

**Severity: High**
**function: openTrading**
**Status: Open**

**Overview:**
The openTrading function permits only the contract owner to activate trading capabilities. Until this function is executed, no investors can buy, sell, or transfer their tokens. This places a high degree of control and centralization in the hands of the contract owner.

```
function openTrading() external onlyOwner {
    require(presaleActive == 3, "LP Not Ready");
    presaleActive = 1;
}
```

**Suggestion**
To reduce centralization and potential manipulation, consider one of the following approaches:
1.Automatically enable trading after a specified condition, such as the completion of a presale, is met.
2.If manual activation is still desired, consider transferring the ownership of the contract to a trustworthy, third-party entity like a certified "PinkSale Safu" developer. This can provide investors with more confidence in the eventual activation of trading capabilities, mitigating concerns of potential bad faith actions by the original owner

# MANUAL TESTING

## Centralization – LP tokens sent to owner

**Severity: High**

**function: openTrading**

**Status: Open**

**Overview:**

**After end of the presale, all LP tokens will be sent to owner**

```
        dexRouter.addLiquidityETH{value: address(this).balance}(
            address(this),
            bal,
            0,
            0,
            owner(),
            block.timestamp
        )
```

**Suggestion**

**Its suggested to lock or burn LP tokens.**

# MANUAL TESTING

## Centralization – Blacklist

**Severity: High**
**function: transferProtection**
**Status: Open**
**Overview:**
**Owner is able to blacklist an arbitrary wallet.**
**Blacklisted wallets wont be able to sell or transfer their tokens**

```
function transferProtection(
address[] calldata _wallets,
uint256 _enabled
  ) external onlyOwner {
for (uint256 i = 0; i < _wallets.length; i++) {
walletProtection[_wallets[i]] = _enabled;
}
  }
```

**Suggestion**
**Implement a more decentralized and automated method for blacklisting bad actors (such as using dead blocks or using maximum wallet/buy/sell/transfer limit)**

# MANUAL TESTING

## Logical – Stuck ETH and tokens

**Severity: High**

**function: transferProtection**

**Status: Open**

**Overview:**

**If presale doesn't reach hardcap, there is no way to withdraw deposited ETH from the contract**

**Suggestion**

**Create a function for allowing participates to withdraw their deposited ETH if presale faild.**

# 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