



Building the Futuristic **Blockchain Ecosystem**

# SECURITY AUDIT REPORT

Arax Network

# TOKEN OVERVIEW

## Risk Findings

Severity	Found
● High	1
● Medium	0
● Low	2
● Informational	2

## Centralization Risks

Owner Privileges	Description
● Can Owner Set Taxes >25% ?	Not Detected
● Owner needs to enable trading ?	Yes, owner needs to enable trades
● Can Owner Disable Trades ?	Not Detected
● Can Owner Mint ?	Not Detected
● Can Owner Blacklist ?	Not Detected
● Can Owner set Max Wallet amount ?	Not Detected
● Can Owner Set Max TX amount ?	Not Detected

# TABLE OF CONTENTS

02	Token Overview	
03	Table of Contents	
04	Overview	
05	Contract Details	
06	Audit Methodology	
07	Vulnerabilities Checklist	
08	Risk Classification	
09	Inheritance Trees	
10	Static analysis	
11	Testnet Version	
12	Manual Review	
18	About Expelee	
19	Disclaimer	

# OVERVIEW

The Expelee team has performed a line-by-line manual analysis and automated review of the smart contract. The smart contract was analysed mainly for common smart contract vulnerabilities, exploits, and manipulation hacks. According to the smart contract audit:

<b>Audit Result</b>	<b>Passed with high risk</b>
<b>KYC Verification</b>	-
<b>Audit Date</b>	<b>20 December 2023</b>

# CONTRACT DETAILS

**Token Address:** 0x47C9b58F75931613E97AeA95141899d6426C4beA

**Name:** Arax Network

**Symbol:** ARAX

**Decimals:** 18

**Network:** BscScan

**Token Type:** BEP-20

**Owner:** 0x78A971D4670f68b93B94F4B58Cb7Eccd939C4558

**Deployer:** 0xC4543C570D6BcF58f9FcB8D868B3560404FC22cf

**Token Supply:** 100000000000

**Checksum:** 27032c616934aeb47e6039f76b20d2b6

**Testnet:**

<https://testnet.bscscan.com/address/0xd46ee741d36851e1355b4a9c5f8bc732ae806a78#code>

# AUDIT METHODOLOGY

## Audit Details

Our comprehensive audit report provides a full overview of the audited system's architecture, smart contract codebase, and details on any vulnerabilities found within the system.

## Audit Goals

The audit goal is to ensure that the project is built to protect investors and users, preventing potentially catastrophic vulnerabilities after launch, that lead to scams and rugpulls.

## Code Quality

Our analysis includes both automatic tests and manual code analysis for the following aspects:

- Exploits
- Back-doors
- Vulnerability
- Accuracy
- Readability

## Tools

- DE
- Open Zeppelin
- Code Analyzer
- Solidity Code
- Compiler
- Hardhat

# VULNERABILITY CHECKS

Design Logic	Passed
Compiler warnings	Passed
Private user data leaks	Passed
Timestamps dependence	Passed
Integer overflow and underflow	Passed
Race conditions & reentrancy. Cross-function race conditions	Passed
Possible delays in data delivery	Passed
Oracle calls	Passed
Front Running	Passed
DoS with Revert	Passed
DoS with block gas limit	Passed
Methods execution permissions	Passed
Economy model	Passed
Impact of the exchange rate on the logic	Passed
Malicious event log	Passed
Scoping and declarations	Passed
Uninitialized storage pointers	Passed
Arithmetic accuracy	Passed
Cross-function race conditions	Passed
Safe Zepplin module	Passed

# RISK CLASSIFICATION

When performing smart contract audits, our specialists look for known vulnerabilities as well as logical and access control issues within the code. The exploitation of these issues by malicious actors may cause serious financial damage to projects that failed to get an audit in time. We categorize these vulnerabilities by the following levels:

## High Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

## Medium Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

## Low Risk

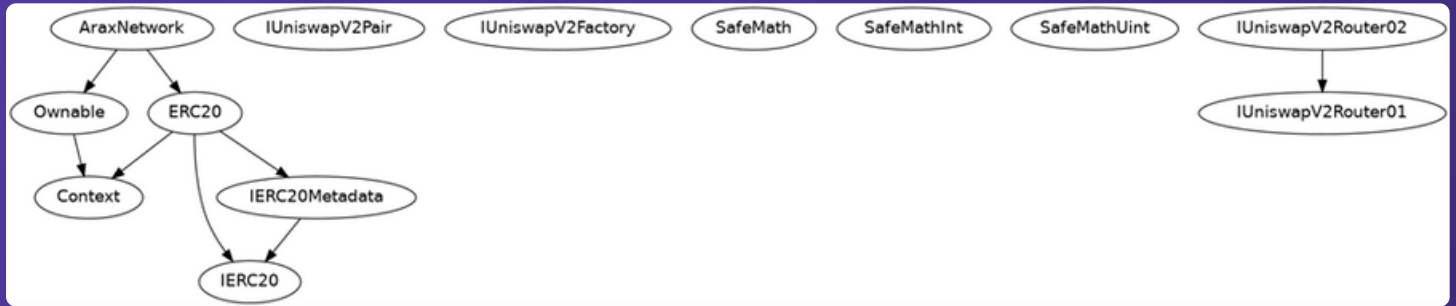
Issues on this level are minor details and warnings that can remain unfixed.

## Informational

Issues on this level are minor details and warnings that can remain unfixed.



# INHERITANCE TREES



# STATIC ANALYSIS

```

INFO:Detectors:
AraxNetwork.constructor().totalSupply (Arax.sol#914) shadows:
  - ERC20.totalSupply (Arax.sol#187) (state variable)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
AraxNetwork.updateBuyFees(uint256) (Arax.sol#988-992) should emit an event for:
  - buyTotalFees = _marketingFee (Arax.sol#998)
AraxNetwork.updateSellFees(uint256) (Arax.sol#994-997) should emit an event for:
  - sellTotalFees = _marketingFee (Arax.sol#995)
AraxNetwork.updateTransferFees(uint256) (Arax.sol#999-1002) should emit an event for:
  - transferTotalFees = _marketingFee (Arax.sol#1000)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
INFO:Detectors:
AraxNetwork.updateMarketingWallet(address).newMarketingWallet (Arax.sol#1022) lacks a zero-check on :
  - marketingWallet = newMarketingWallet (Arax.sol#1024)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in AraxNetwork._transfer(address,address,uint256) (Arax.sol#1031-1101):
  External calls:
    - swapBack() (Arax.sol#1060)
      - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (Arax.sol#1113-1119)
      - (success,None) = address(marketingWallet).call{value: address(this).balance}() (Arax.sol#1132)
  External calls sending eth:
    - swapBack() (Arax.sol#1060)
      - (success,None) = address(marketingWallet).call{value: address(this).balance}() (Arax.sol#1132)
  Event emitted after the call(s):
    - Transfer(sender,recipient,amount) (Arax.sol#369)
      - super._transfer(from,address(this),fees) (Arax.sol#1094)
    - Transfer(sender,recipient,amount) (Arax.sol#369)
      - super._transfer(from,to,amount) (Arax.sol#1100)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
Context._msgData() (Arax.sol#14-17) is never used and should be removed
ERC20._burn(address,uint256) (Arax.sol#402-410) is never used and should be removed
SafeMath.mod(uint256,uint256) (Arax.sol#579-581) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (Arax.sol#595-598) is never used and should be removed
SafeMath.sub(uint256,uint256) (Arax.sol#486-488) is never used and should be removed
SafeMathInt.abs(int256) (Arax.sol#703-706) is never used and should be removed
SafeMathInt.add(int256,int256) (Arax.sol#694-698) is never used and should be removed
SafeMathInt.div(int256,int256) (Arax.sol#674-680) is never used and should be removed
SafeMathInt.mul(int256,int256) (Arax.sol#662-669) is never used and should be removed
SafeMathInt.sub(int256,int256) (Arax.sol#685-689) is never used and should be removed
SafeMathInt.toInt256Safe(int256) (Arax.sol#709-712) is never used and should be removed
SafeMathUInt.toInt256Safe(uint256) (Arax.sol#716-720) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version0.8.9 (Arax.sol#7) allows old versions
solc-0.8.9 is not recommended for deployment

```

```

INFO:Detectors:
Low level call in AraxNetwork.swapBack() (Arax.sol#1123-1134):
  - (success,None) = address(marketingWallet).call{value: address(this).balance}() (Arax.sol#1132)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (Arax.sol#35) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (Arax.sol#36) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (Arax.sol#63) is not in mixedCase
Function IUniswapV2Router01.WETH() (Arax.sol#726) is not in mixedCase
Event AraxNetwork.marketingWalletUpdated(address,address) (Arax.sol#896) is not in CapWords
Parameter AraxNetwork.updateBuyFees(uint256)._marketingFee (Arax.sol#988) is not in mixedCase
Parameter AraxNetwork.updateSellFees(uint256)._marketingFee (Arax.sol#994) is not in mixedCase
Parameter AraxNetwork.updateTransferFees(uint256)._marketingFee (Arax.sol#999) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (Arax.sol#16)" inContext (Arax.sol#49-10)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,address,uint256).amountADesired (Arax.sol#731) is too similar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,address,uint256).amountBDesired (Arax.sol#732)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
SafeMathInt.MAX_INT256 (Arax.sol#657) is never used in SafeMathInt (Arax.sol#655-713)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable
INFO:Detectors:
AraxNetwork._decimals (Arax.sol#876) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
INFO:Slither:Arax.sol analyzed (13 contracts with 93 detectors), 34 result(s) found

```

# TESTNET VERSION

## 1- Approve (passed):

<https://testnet.bscscan.com/tx/0x88fbc0f66ac56539657945579d11bb8deb53523ce3ce77471ae1c8035f5d4e05>

## 2- Increase Allowance (passed):

<https://testnet.bscscan.com/tx/0x3574b37cd626eec510a7358b9917396d25f8e62008c217484f5fee117a0817c7>

## 3- Decrease Allowance (passed):

<https://testnet.bscscan.com/tx/0x04e742c4cfec517aa0c6b828a968e53a221833dfa9e20ee74a0426b0662489ba>

## 4- Exclude From Fees (passed):

<https://testnet.bscscan.com/tx/0x3b73a36412254d15ad8e85a7b08b015766cd019c3f19eb758e21505d2df7edca>

## 5- Transfer From (passed):

<https://testnet.bscscan.com/tx/0x8d4456228e94e2d682d4f2e236fb0d464479a07d9082e32da62325c1479f486f>

# MANUAL REVIEW

## Severity Criteria

Expelee assesses the severity of disclosed vulnerabilities according to methodology based on OWASP standarts.

Vulnerabilities are dividend into three primary risk categorieis:

High

Medium

Low

High-level considerations for vulnerabilities span the following key areas when conducting assessments:

- Malicious input handling
- Escalation of privileges
- Arithmetic
- Gas use

Overall Risk Severity				
Impact	HIGH	Medium	High	Critical
	MEDIUM	Low	Medium	High
	LOW	Note	Low	Medium
		LOW	MEDIUM	HIGH
	Likelihood			

# HIGH RISK FINDING

## Centralization – Enabling Trades

Severity: High

function: EnableTrading

Status: Open

### Overview:

The EnableTrading 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 enableTrading() external onlyOwner {  
    tradingActive = true;  
    swapEnabled = true;  
}
```

### 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 give investors more confidence in the eventual activation of trading capabilities, mitigating concerns of potential bad-faith actions by the original owner.

# LOW RISK FINDING

## Centralization – Missing Events

Severity: **Low**

subject: Missing Events

Status: Open

### Overview:

They serve as a mechanism for emitting and recording data onto the blockchain, making it transparent and easily accessible.

```
function updateBuyFees(uint256 _marketingFee) external onlyOwner {  
    buyTotalFees = _marketingFee;  
    require(buyTotalFees <= 10, "Must keep fees at 10% or less");  
}  
function updateSellFees(uint256 _marketingFee) external onlyOwner {  
    sellTotalFees = _marketingFee;  
    require(sellTotalFees <= 10, "Must keep fees at 10% or less");  
}  
function updateTransferFees(uint256 _marketingFee) external  
onlyOwner {  
    transferTotalFees = _marketingFee ;  
    require(transferTotalFees <= 10, "Must keep fees at 10% or less");  
}  
function enableTrading() external onlyOwner {  
    tradingActive = true;  
    swapEnabled = true;  
}
```



# LOW RISK FINDING

## Centralization – Missing Zero Address

Severity: Low

Subject: Zero Check

Status: Open

### Overview:

functions can take a zero address as a parameter (0x00000...). If a function parameter of address type is not properly validated by checking for zero addresses, there could be serious consequences for the contract's functionality.

```
function updateMarketingWallet(address  
newMarketingWallet) external onlyOwner {  
    emit  
marketingWalletUpdated(newMarketingWallet,  
marketingWallet);  
    marketingWallet = newMarketingWallet;  
}
```

# INFORMATIONAL & OPTIMIZATIONS

## Optimization

Severity: Optimization

subject: Remove unused code.

Status: Open

### Overview:

Unused variables are allowed in Solidity, and they do. not pose a direct security issue. It is the best practice. though to avoid them

```
function _msgData() internal view virtual returns (bytes calldata) {  
    this; // silence state mutability warning without generating bytecode  
    - see https://github.com/ethereum/solidity/issues/2691  
    return msg.data;  
}  
}
```



# INFORMATIONAL & OPTIMIZATIONS

## Optimization

Severity: Informational

subject: Remove Safe Math

Status: Open

Line: 458 – 599

### Overview:

compiler version above 0.8.0 can control arithmetic overflow/underflow, It is recommended to remove the unwanted code to avoid high gas fees.

# ABOUT EXPELEE

Expelee is a product-based aspirational Web3 start-up. Coping up with numerous solutions for blockchain security and constructing a Web3 ecosystem from deal making platform to developer hosting open platform, while also developing our own commercial and sustainable blockchain.

 [www.expelee.com](http://www.expelee.com)

 [expeleeofficial](https://twitter.com/expeleeofficial)

 [expelee](https://medium.com/expelee)

 [Expelee](https://t.me/Expelee)

 [expelee](https://in.linkedin.com/company/expelee)

 [expelee\\_official](https://www.instagram.com/expelee_official)

 [expelee-co](https://github.com/expelee-co)

# expelee

Building the Futuristic **Blockchain Ecosystem**

# 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 guarantess against the sale of team tokens or the removal of liquidity by the project audited in this document.

Always do your own research and project yourselves from being scammed. The Expelee 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 Expelee receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Alway 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 Expelee team disclaims any liability for the resulting losses.

The logo for Expelee, featuring the word "expelee" in a stylized font. The "ex" is in white, and "pelee" is in orange. The letters are bold and modern.

Building the Futuristic **Blockchain Ecosystem**