expelee

Building the Futuristic Blockchain Ecosystem

SECURITY AUDIT REPORT

NexusDAO

TABLE OF CONTENTS

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:

Audit Result	**Passed**
KYC Verification	**Done**
Audit Date	**15 May 2023**

CONTRACT DETAILS

Token Name: Nexus DAO

Symbol: nxsDAO

Network: Binance Smart Chain

Language: Solidity

Contract Address:
0x4EA82C3f321adC2Da81754DB288C6B5FD8a22645

Total Supply: 1000000000

Contract SHA-256 Checksum: -

Owner's Wallet:
0xf97cAb5742e1052f2BDCcBAC2527c3Fceb9f9284

Deployer's Wallet:
0xf97cAb5742e1052f2BDCcBAC2527c3Fceb9f9284

Testnet:
https://testnet.bscscan.com/address/0xccba1299e605f349
72e3ea197eb8289d1e62954e

OWNER PRIVILEGES

- Owner can exclude/include accounts from rewards
- Owner can exclude accounts from fees
- Owner can change fee percentages max 10%
- Trading must be enabled by the owner
- Owner can change the swap tokens at amount within reasonable limit
- Owner can change swap setting
- Owner can withdraw any token(except native token) from the contract
- Owner can change the marketing wallet
- Owner can enable/disable wallet to wallet transfer without fee

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 acces 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.

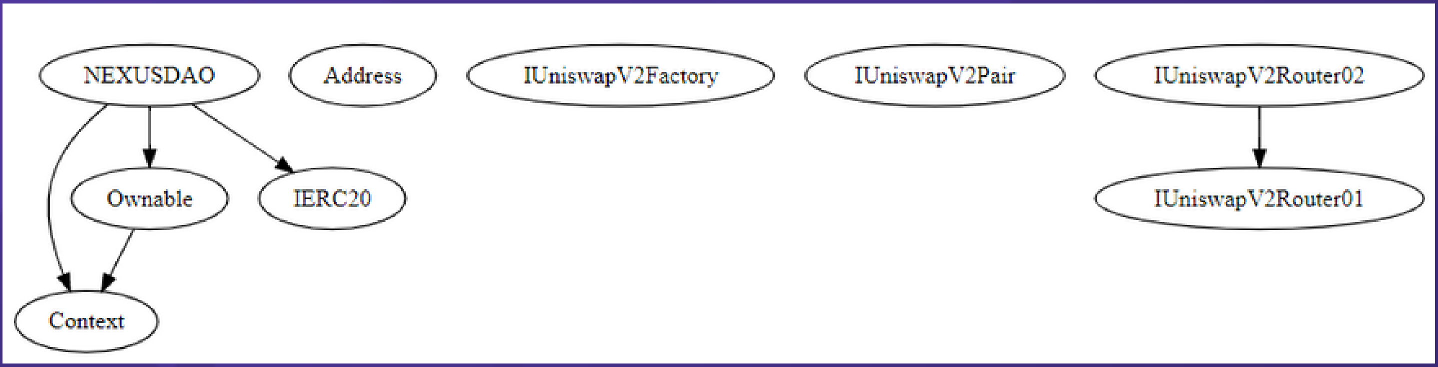
Low Risk

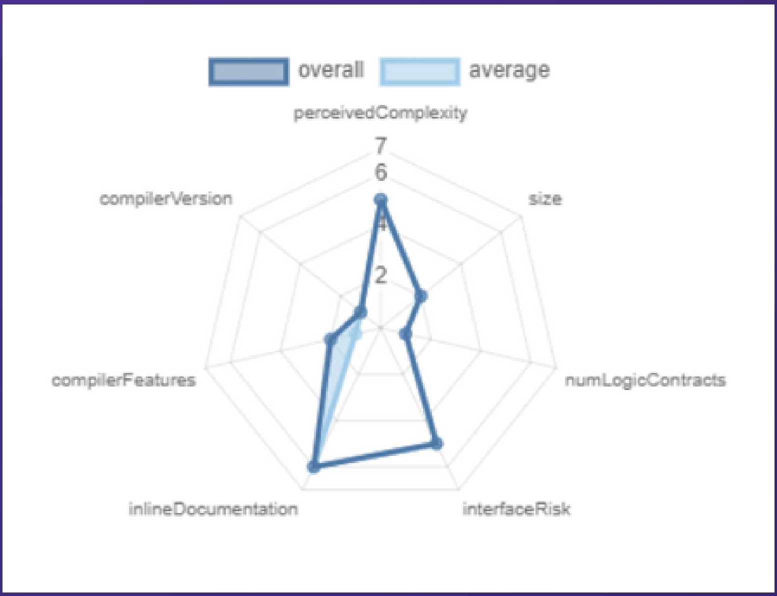
Issues on this level are minor details and warning that can remain unfixed.

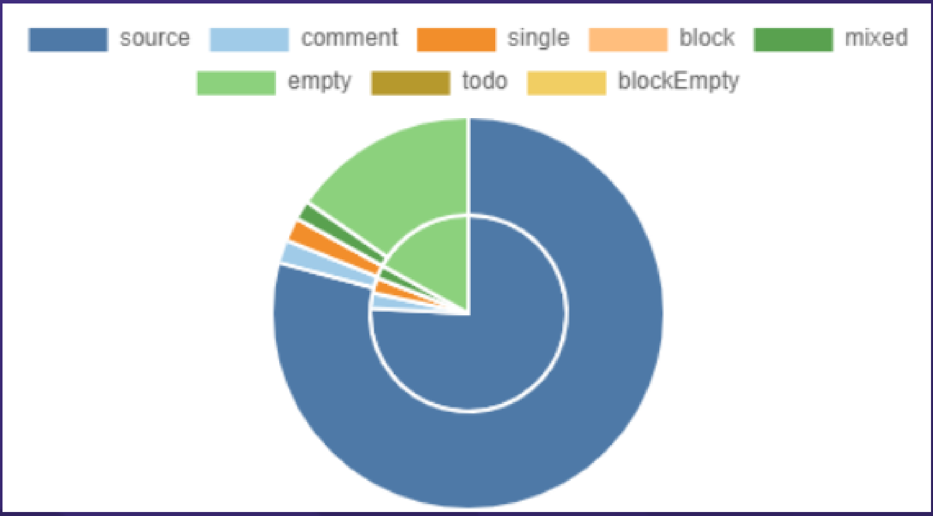
Informational

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

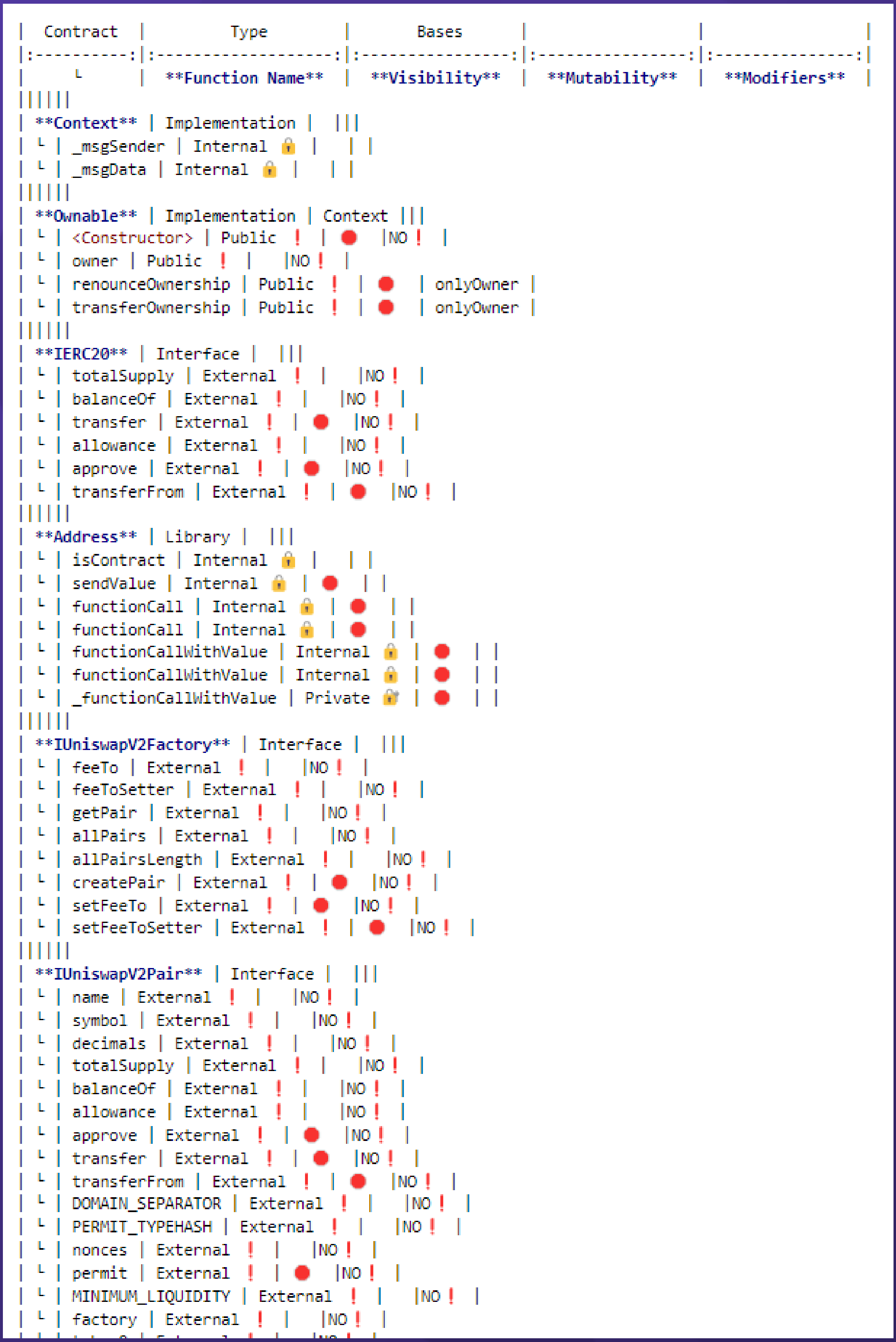
INHERITANCE TREES





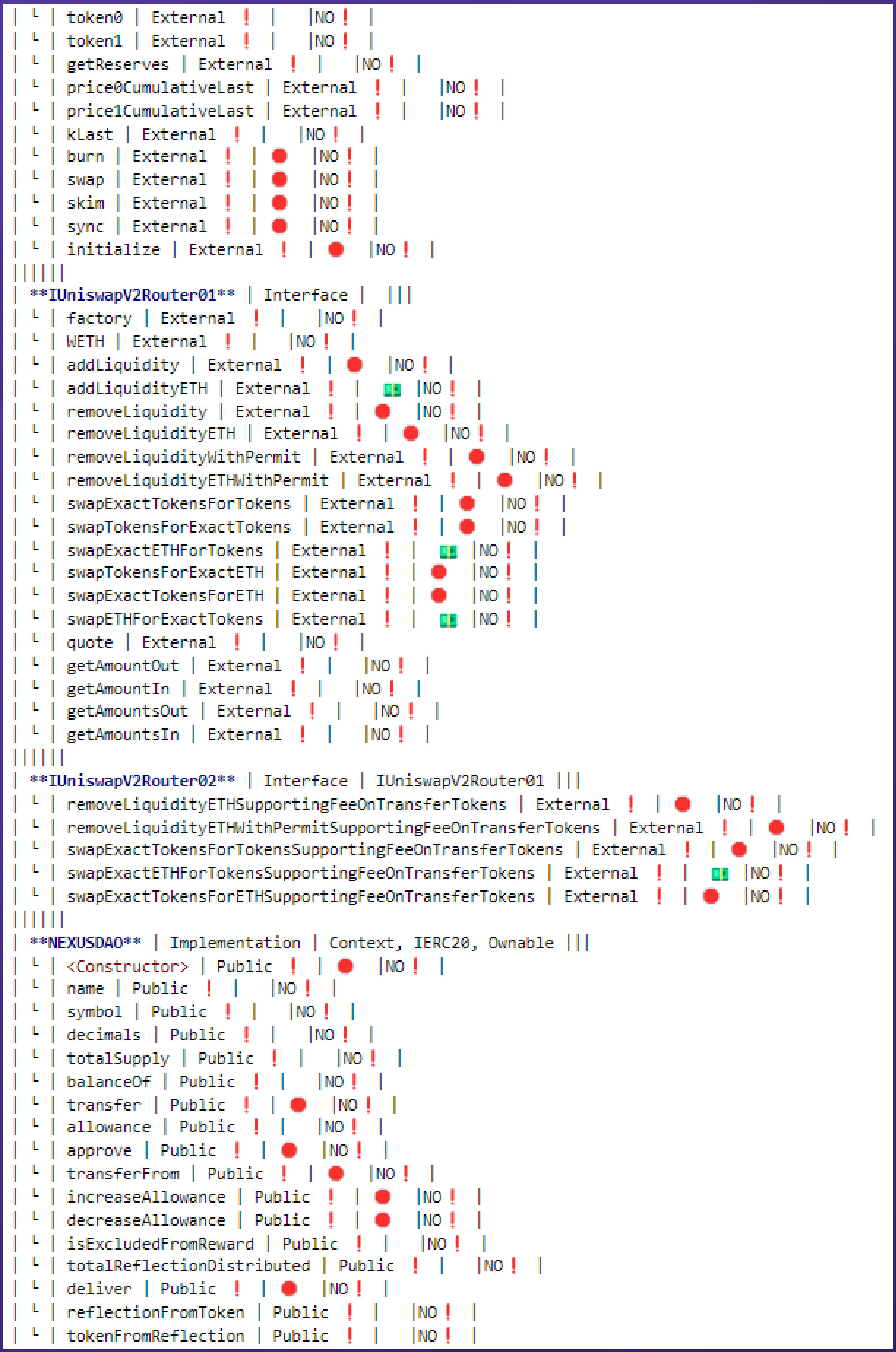


FUNCTION DETAILS



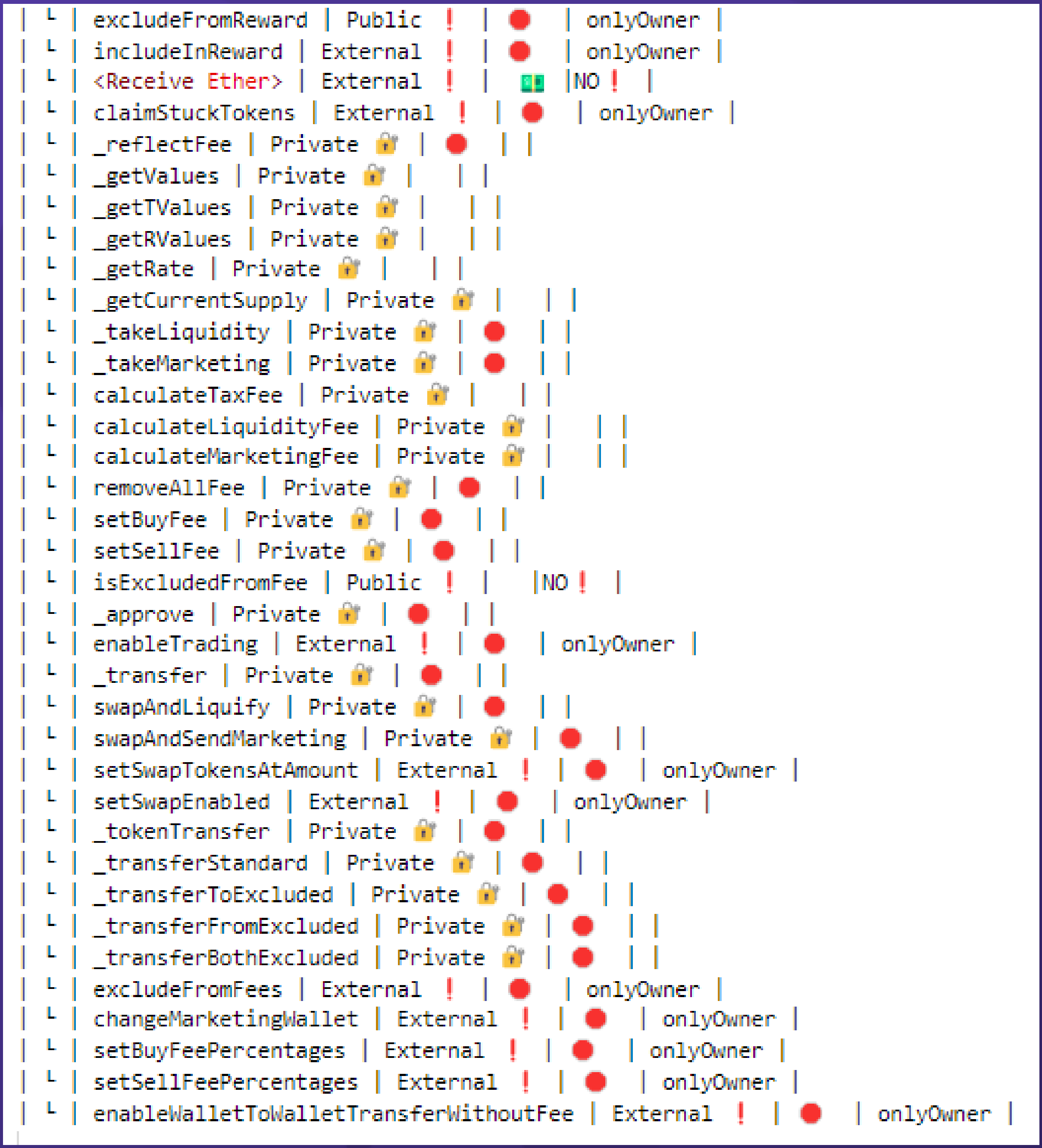
```
|  Contract  |        Type        |       Bases      |                  |                 |
|:----------:|:-------------------:|:----------------:|:----------------:|:---------------:|
|     └      |  **Function Name**  |  **Visibility**  |  **Mutability**  |  **Modifiers**  |
||||||
| **Context** | Implementation |  |||
| └ | _msgSender | Internal 🔒 |   | |
| └ | _msgData | Internal 🔒 |   | |
||||||
| **Ownable** | Implementation | Context |||
| └ | <Constructor> | Public ❗️ | 🛑  |NO❗️ |
| └ | owner | Public ❗️ |   |NO❗️ |
| └ | renounceOwnership | Public ❗️ | 🛑  | onlyOwner |
| └ | transferOwnership | Public ❗️ | 🛑  | onlyOwner |
||||||
| **IERC20** | Interface |  |||
| └ | totalSupply | External ❗️ |   |NO❗️ |
| └ | balanceOf | External ❗️ |   |NO❗️ |
| └ | transfer | External ❗️ | 🛑  |NO❗️ |
| └ | allowance | External ❗️ |   |NO❗️ |
| └ | approve | External ❗️ | 🛑  |NO❗️ |
| └ | transferFrom | External ❗️ | 🛑  |NO❗️ |
||||||
| **Address** | Library |  |||
| └ | isContract | Internal 🔒 |   | |
| └ | sendValue | Internal 🔒 | 🛑  | |
| └ | functionCall | Internal 🔒 | 🛑  | |
| └ | functionCall | Internal 🔒 | 🛑  | |
| └ | functionCallWithValue | Internal 🔒 | 🛑  | |
| └ | functionCallWithValue | Internal 🔒 | 🛑  | |
| └ | _functionCallWithValue | Private 🔐 | 🛑  | |
||||||
| **IUniswapV2Factory** | Interface |  |||
| └ | feeTo | External ❗️ |   |NO❗️ |
| └ | feeToSetter | External ❗️ |   |NO❗️ |
| └ | getPair | External ❗️ |   |NO❗️ |
| └ | allPairs | External ❗️ |   |NO❗️ |
| └ | allPairsLength | External ❗️ |   |NO❗️ |
| └ | createPair | External ❗️ | 🛑  |NO❗️ |
| └ | setFeeTo | External ❗️ | 🛑  |NO❗️ |
| └ | setFeeToSetter | External ❗️ | 🛑  |NO❗️ |
||||||
| **IUniswapV2Pair** | Interface |  |||
| └ | name | External ❗️ |   |NO❗️ |
| └ | symbol | External ❗️ |   |NO❗️ |
| └ | decimals | External ❗️ |   |NO❗️ |
| └ | totalSupply | External ❗️ |   |NO❗️ |
| └ | balanceOf | External ❗️ |   |NO❗️ |
| └ | allowance | External ❗️ |   |NO❗️ |
| └ | approve | External ❗️ | 🛑  |NO❗️ |
| └ | transfer | External ❗️ | 🛑  |NO❗️ |
| └ | transferFrom | External ❗️ | 🛑  |NO❗️ |
| └ | DOMAIN_SEPARATOR | External ❗️ |   |NO❗️ |
| └ | PERMIT_TYPEHASH | External ❗️ |   |NO❗️ |
| └ | nonces | External ❗️ |   |NO❗️ |
| └ | permit | External ❗️ | 🛑  |NO❗️ |
| └ | MINIMUM_LIQUIDITY | External ❗️ |   |NO❗️ |
| └ | factory | External ❗️ |   |NO❗️ |
```

FUNCTION DETAILS



```
| └ | token0 | External ❗ |   |NO❗ |
| └ | token1 | External ❗ |   |NO❗ |
| └ | getReserves | External ❗ |   |NO❗ |
| └ | price0CumulativeLast | External ❗ |   |NO❗ |
| └ | price1CumulativeLast | External ❗ |   |NO❗ |
| └ | kLast | External ❗ |   |NO❗ |
| └ | burn | External ❗ | 🛑  |NO❗ |
| └ | swap | External ❗ | 🛑  |NO❗ |
| └ | skim | External ❗ | 🛑  |NO❗ |
| └ | sync | External ❗ | 🛑  |NO❗ |
| └ | initialize | External ❗ | 🛑  |NO❗ |
||||||
| **IUniswapV2Router01** | Interface |  |||
| └ | factory | External ❗ |   |NO❗ |
| └ | WETH | External ❗ |   |NO❗ |
| └ | addLiquidity | External ❗ | 🛑  |NO❗ |
| └ | addLiquidityETH | External ❗ | 💵 |NO❗ |
| └ | removeLiquidity | External ❗ | 🛑  |NO❗ |
| └ | removeLiquidityETH | External ❗ | 🛑  |NO❗ |
| └ | removeLiquidityWithPermit | External ❗ | 🛑  |NO❗ |
| └ | removeLiquidityETHWithPermit | External ❗ | 🛑  |NO❗ |
| └ | swapExactTokensForTokens | External ❗ | 🛑  |NO❗ |
| └ | swapTokensForExactTokens | External ❗ | 🛑  |NO❗ |
| └ | swapExactETHForTokens | External ❗ | 💵 |NO❗ |
| └ | swapTokensForExactETH | External ❗ | 🛑  |NO❗ |
| └ | swapExactTokensForETH | External ❗ | 🛑  |NO❗ |
| └ | swapETHForExactTokens | External ❗ | 💵 |NO❗ |
| └ | quote | External ❗ |   |NO❗ |
| └ | getAmountOut | External ❗ |   |NO❗ |
| └ | getAmountIn | External ❗ |   |NO❗ |
| └ | getAmountsOut | External ❗ |   |NO❗ |
| └ | getAmountsIn | External ❗ |   |NO❗ |
||||||
| **IUniswapV2Router02** | Interface | IUniswapV2Router01 |||
| └ | removeLiquidityETHSupportingFeeOnTransferTokens | External ❗ | 🛑  |NO❗ |
| └ | removeLiquidityETHWithPermitSupportingFeeOnTransferTokens | External ❗ | 🛑  |NO❗ |
| └ | swapExactTokensForTokensSupportingFeeOnTransferTokens | External ❗ | 🛑  |NO❗ |
| └ | swapExactETHForTokensSupportingFeeOnTransferTokens | External ❗ | 💵 |NO❗ |
| └ | swapExactTokensForETHSupportingFeeOnTransferTokens | External ❗ | 🛑  |NO❗ |
||||||
| **NEXUSDAO** | Implementation | Context, IERC20, Ownable |||
| └ | <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❗ |
| └ | isExcludedFromReward | Public ❗ |   |NO❗ |
| └ | totalReflectionDistributed | Public ❗ |   |NO❗ |
| └ | deliver | Public ❗ | 🛑  |NO❗ |
| └ | reflectionFromToken | Public ❗ |   |NO❗ |
| └ | tokenFromReflection | Public ❗ |   |NO❗ |
```

FUNCTION DETAILS



```
| └ | excludeFromReward | Public ❗ | 🛑  | onlyOwner |
| └ | includeInReward | External ❗ | 🛑  | onlyOwner |
| └ | <Receive Ether> | External ❗ |  💵 |NO ❗ |
| └ | claimStuckTokens | External ❗ | 🛑  | onlyOwner |
| └ | _reflectFee | Private 🔐 | 🛑  | |
| └ | _getValues | Private 🔐 |   | |
| └ | _getTValues | Private 🔐 |   | |
| └ | _getRValues | Private 🔐 |   | |
| └ | _getRate | Private 🔐 |   | |
| └ | _getCurrentSupply | Private 🔐 |   | |
| └ | _takeLiquidity | Private 🔐 | 🛑  | |
| └ | _takeMarketing | Private 🔐 | 🛑  | |
| └ | calculateTaxFee | Private 🔐 |   | |
| └ | calculateLiquidityFee | Private 🔐 |   | |
| └ | calculateMarketingFee | Private 🔐 |   | |
| └ | removeAllFee | Private 🔐 | 🛑  | |
| └ | setBuyFee | Private 🔐 | 🛑  | |
| └ | setSellFee | Private 🔐 | 🛑  | |
| └ | isExcludedFromFee | Public ❗ |   |NO ❗ |
| └ | _approve | Private 🔐 | 🛑  | |
| └ | enableTrading | External ❗ | 🛑  | onlyOwner |
| └ | _transfer | Private 🔐 | 🛑  | |
| └ | swapAndLiquify | Private 🔐 | 🛑  | |
| └ | swapAndSendMarketing | Private 🔐 | 🛑  | |
| └ | setSwapTokensAtAmount | External ❗ | 🛑  | onlyOwner |
| └ | setSwapEnabled | External ❗ | 🛑  | onlyOwner |
| └ | _tokenTransfer | Private 🔐 | 🛑  | |
| └ | _transferStandard | Private 🔐 | 🛑  | |
| └ | _transferToExcluded | Private 🔐 | 🛑  | |
| └ | _transferFromExcluded | Private 🔐 | 🛑  | |
| └ | _transferBothExcluded | Private 🔐 | 🛑  | |
| └ | excludeFromFees | External ❗ | 🛑  | onlyOwner |
| └ | changeMarketingWallet | External ❗ | 🛑  | onlyOwner |
| └ | setBuyFeePercentages | External ❗ | 🛑  | onlyOwner |
| └ | setSellFeePercentages | External ❗ | 🛑  | onlyOwner |
| └ | enableWalletToWalletTransferWithoutFee | External ❗ | 🛑  | onlyOwner |
```

MANUAL REVIEW

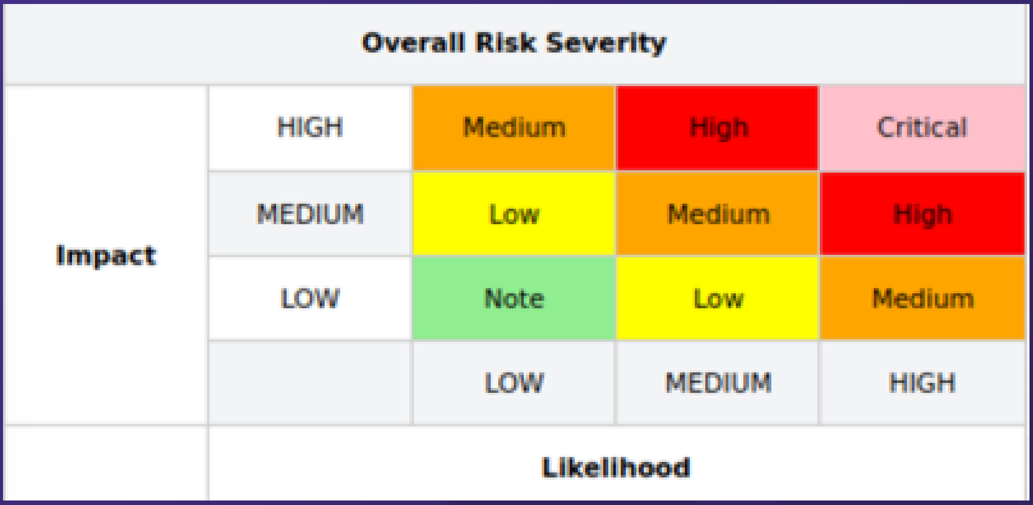
Severity Criteria

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

Vulnerabilities are dividend into three primary risk categroies:
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			

FINDINGS

Findings	Severity	Found
High Risk	● High	0
Medium Risk	● Medium	1
Low Risk	● Low	6
Suggestion & discussion	● Informational	0
Gas Optimizations	● Gas Opt.	0

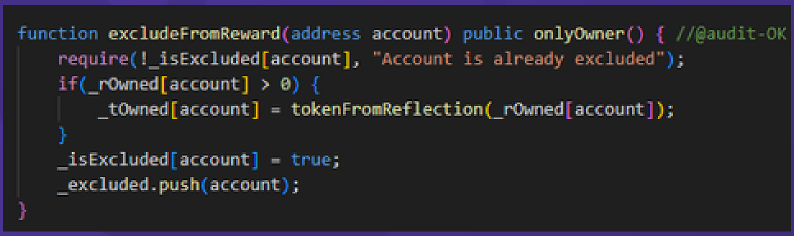
MEDIUM RISK FINDING

Owner can exclude/include accounts from rewards

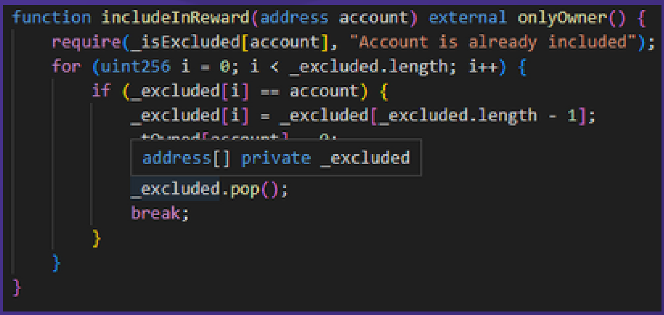
Severity : Medium

Overview

Function that allows the owner of the contract to exclude an address from receiving dividends



```
function excludeFromReward(address account) public onlyOwner() { //@audit-OK
    require(!_isExcluded[account], "Account is already excluded");
    if(_rOwned[account] > 0) {
        _tOwned[account] = tokenFromReflection(_rOwned[account]);
    }
    _isExcluded[account] = true;
    _excluded.push(account);
}
```



```
function includeInReward(address account) external onlyOwner() {
    require(_isExcluded[account], "Account is already included");
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (_excluded[i] == account) {
            _excluded[i] = _excluded[_excluded.length - 1];
            _tOwned[account] = 0;
             address[] private _excluded
            _excluded.pop();
            break;
        }
    }
}
```

Recommendation

It is recommended to add additional access control measures, such as multi-factor authentication or time-based restrictions, to limit the number of authorized users who can call these functions. The contract owner account is well secured and only accessible by authorized parties.

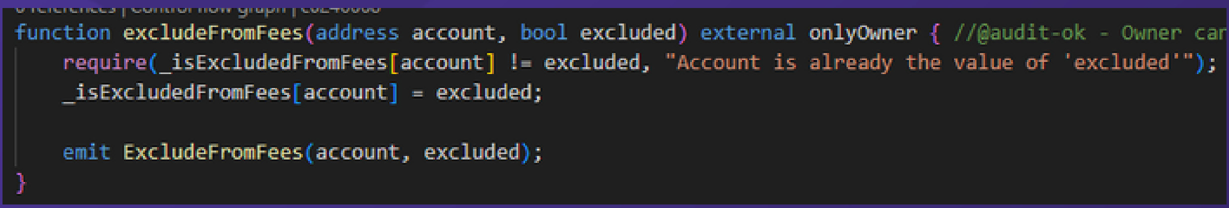
LOW RISK FINDING

Owner can exclude accounts from fees

Severity : Low

Overview

Excludes/Includes an address from the collection of fees



```
function excludeFromFees(address account, bool excluded) external onlyOwner { //@audit-ok - Owner car
    require(_isExcludedFromFees[account] != excluded, "Account is already the value of 'excluded'");
    _isExcludedFromFees[account] = excluded;

    emit ExcludeFromFees(account, excluded);
}
```

Recommendation

It is recommended to add additional access control measures, such as multi-factor authentication or time-based restrictions, to limit the number of authorized users who can call these functions. The contract owner account is well secured and only accessible by authorized parties.

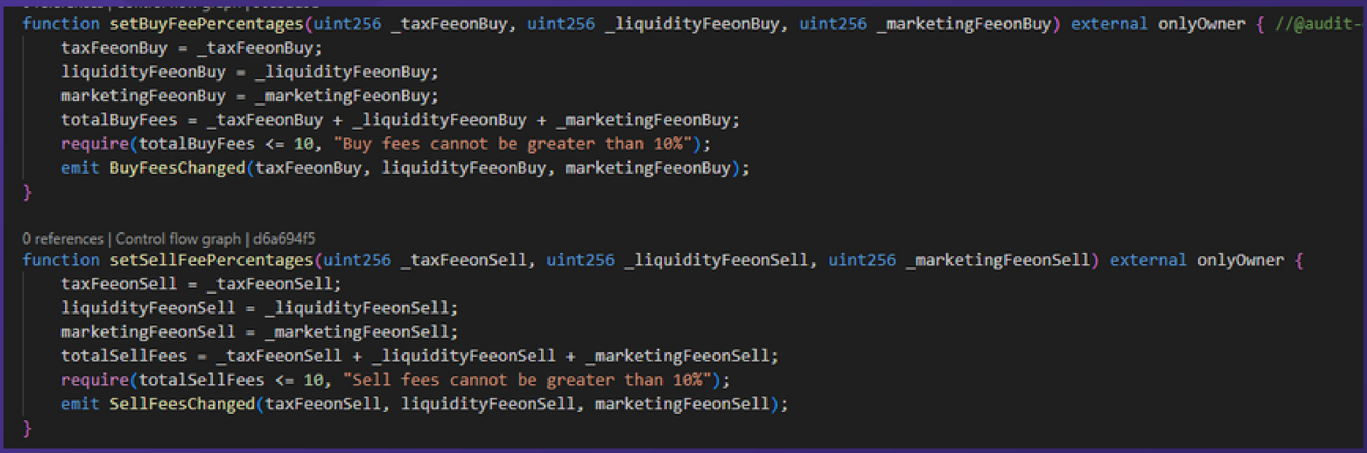
LOW RISK FINDING

Owner can change fee percentages max 10%

Severity : Low

Overview

Functions that allows the owner of the contract to update the buy/sell fees of the contract. These functions assumes that the input parameters are valid and do not exceed the maximum limit of 10%



```
function setBuyFeePercentages(uint256 _taxFeeonBuy, uint256 _liquidityFeeonBuy, uint256 _marketingFeeonBuy) external onlyOwner { //@audit-
    taxFeeonBuy = _taxFeeonBuy;
    liquidityFeeonBuy = _liquidityFeeonBuy;
    marketingFeeonBuy = _marketingFeeonBuy;
    totalBuyFees = _taxFeeonBuy + _liquidityFeeonBuy + _marketingFeeonBuy;
    require(totalBuyFees <= 10, "Buy fees cannot be greater than 10%");
    emit BuyFeesChanged(taxFeeonBuy, liquidityFeeonBuy, marketingFeeonBuy);
}

0 references | Control flow graph | d6a694f5
function setSellFeePercentages(uint256 _taxFeeonSell, uint256 _liquidityFeeonSell, uint256 _marketingFeeonSell) external onlyOwner {
    taxFeeonSell = _taxFeeonSell;
    liquidityFeeonSell = _liquidityFeeonSell;
    marketingFeeonSell = _marketingFeeonSell;
    totalSellFees = _taxFeeonSell + _liquidityFeeonSell + _marketingFeeonSell;
    require(totalSellFees <= 10, "Sell fees cannot be greater than 10%");
    emit SellFeesChanged(taxFeeonSell, liquidityFeeonSell, marketingFeeonSell);
}
```

Recommendation

It is recommended to add additional access control measures, such as multi-factor authentication or time-based restrictions, to limit the number of authorized users who can call these functions. The contract owner account is well secured and only accessible by authorized parties.

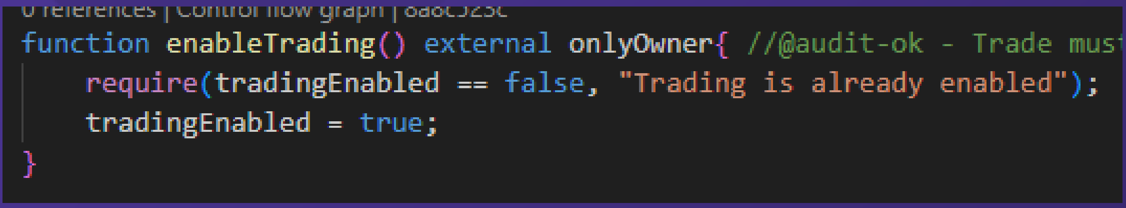
LOW RISK FINDING

Trading must be enabled by the owner

Severity : Low

Overview

Function enables trading by setting the **tradingEnabled** true



```
0 references | Control flow graph | 8a8c523c
function enableTrading() external onlyOwner{ //@audit-ok - Trade mus
    require(tradingEnabled == false, "Trading is already enabled");
    tradingEnabled = true;
}
```

Recommendation

It is recommended to add additional access control measures, such as multi-factor authentication or time-based restrictions, to limit the number of authorized users who can call these functions. The contract owner account is well secured and only accessible by authorized parties.

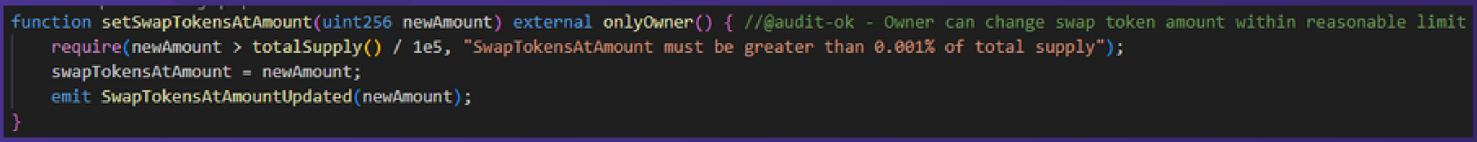
LOW RISK FINDING

Owner can change the swap tokens at amount within reasonable limit

Severity : Low

Overview

setSwapTokensAtAmount function allows the owner to set the minimum number of tokens required to trigger an automatic swap.



```
function setSwapTokensAtAmount(uint256 newAmount) external onlyOwner() { //@audit-ok - Owner can change swap token amount within reasonable limit
    require(newAmount > totalSupply() / 1e5, "SwapTokensAtAmount must be greater than 0.001% of total supply");
    swapTokensAtAmount = newAmount;
    emit SwapTokensAtAmountUpdated(newAmount);
}
```

Recommendation

It's important to ensure that the new **swapTokensAtAmount** value is reasonable and will not adversely affect the functioning of the token or any associated systems.

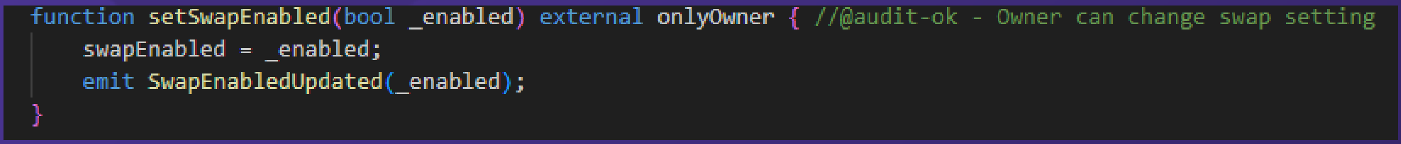
LOW RISK FINDING

Owner can change swap setting

Severity : Low

Overview

Function allows the contract owner to enable or disable the automatic swapping.



```
function setSwapEnabled(bool _enabled) external onlyOwner { //@audit-ok - Owner can change swap setting
    swapEnabled = _enabled;
    emit SwapEnabledUpdated(_enabled);
}
```

Recommendation

It is recommended to ensure that the contract owner account is well secured and only accessible by authorized parties.

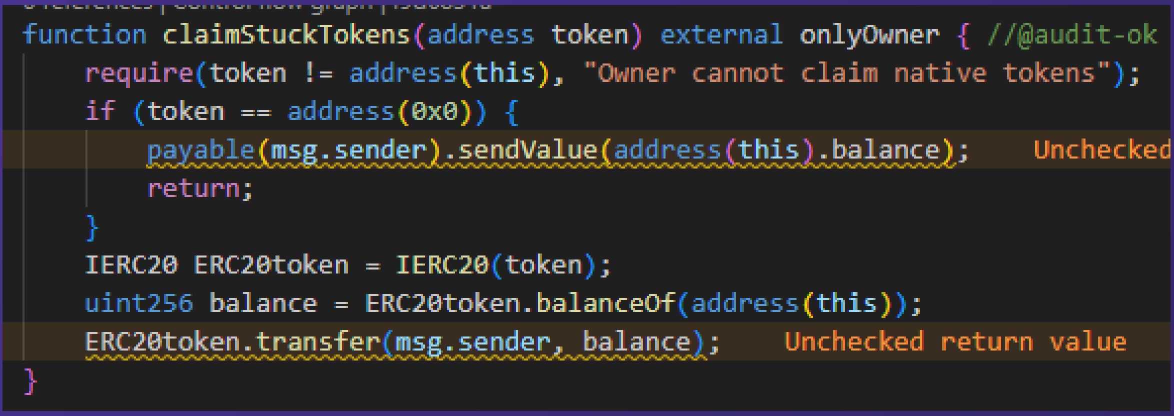
LOW RISK FINDING

Owner can withdraw any token(except native token) from the contract

Severity : Low

Overview

claimStuckTokens function allows the contract owner to recover any ERC20 tokens or BNB that were mistakenly sent to the contract's address. There arerequire statement to prevent the owner from accidentally claiming the native token.



```
function claimStuckTokens(address token) external onlyOwner { //@audit-ok
    require(token != address(this), "Owner cannot claim native tokens");
    if (token == address(0x0)) {
        payable(msg.sender).sendValue(address(this).balance);    Unchecked
        return;
    }
    IERC20 ERC20token = IERC20(token);
    uint256 balance = ERC20token.balanceOf(address(this));
    ERC20token.transfer(msg.sender, balance);    Unchecked return value
}
```

Recommendation

It is generally considered safe for a contract owner to claim stuck tokens, but it's important to ensure that the owner is not abusing this function to steal tokens. In this implementation, there is a require statement that ensures that the **owner cannot claim the native token** of the blockchain on which the contract is deployed.

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

expeleeofficial	expelee
Expelee	expelee
expelee_official	expelee-co

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.

expelee

Building the Futuristic Blockchain Ecosystem