SecureWise

AUDIT REPORT

SecureWise

SMART CONTRACT AUDIT

METACUBEZ

https://github.com/securewise
https://t.me/securewise
https://securewise.info/

Table of Contents

Disclaimer

SecureWise provides the smart contract audit of solidity. Audit and report are for informational purposes only and not, nor should be considered, as an endorsement to engage with, invest in, participate, provide an incentive, or disapprove, criticise, discourage, or purport to provide an opinion on any particular project or team.

This audit report doesn't provide any warranty or guarantee regarding the nature of the technology analysed. These reports, in no way, provide investment advice, nor should be used as investment advice of any sort. Investors must always do their own research and manage their risk.

DISCLAIMER: By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and SecureWise and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) SecureWise owe no duty of care towards you or any other person, nor does SecureWise make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and SecureWise hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, SecureWise hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against SecureWise, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report.

Overview

Token Name: METACUBEZ (MCubez)

Methodology: Automated Analysis, Manual Code Review

Language: Solidity

Contract Address: 0x4352e3CBBa64Ba07401b8928885528dEaE66C2c3

ContractLink: https://bscscan.com/address/0x4352e3CBBa64Ba07401b8928885528dEaE66C2c3

Network: Binance Smart Chain (BSC)

Decimals: 4

Supply: 200.000.000

Website: https://metacubez.io/

Twitter: https://twitter.com/metacubez

Telegram: https://t.me/metacubezchat

Report Date: August 31, 2022

Quick Result

SecureWise has applied the automated and manual analysis of Smart Contract and were reviewed for common contract vulnerabilities and centralized exploits

Owner Privileges

- ⚠ Auto liquidity is going to an externally owned account
- ⚠ The owner can exclude accounts from rewards
- ⚠ The owner can exclude accounts from fees
- ⚠ Current fees 7% and it can not be set
- ⚠ The owner can change swap settings

METACUBEZ (MCubez) has succesfully **PASSED** the smart contract audit with **MEDIUM** AND **LOW** severity issue

Auditing Approach and Methodologies

SecureWise has performed starting with analyzing the code, issues, code quality, and libraries. Reviewed line-by-line by our team. Finding any potential issue like race conditions, transaction-ordering dependence, timestamp dependence, and denial of service attacks.

Methodology

- Understanding the size, scope and functionality of your project's source code
- Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
- Testing and automated analysis of the Smart Contract to determine proper logic has been followed throughout the whole process
- Deploying the code on testnet using multiple live test
- Analyzing a program to determine the specific input that causes different parts of a program to execute its functions.
- Checking whether all the libraries used in the code are on the latest version.

Goals

Smart Contract System is secure, resilient and working according to the specifications and without any vulnerabilities.

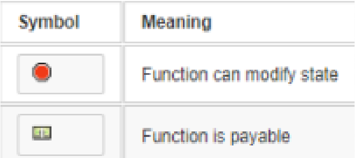
Risk Classification

High: Exploits, vulnerabilities or errors that will certainly or probabilistically lead towards loss of funds, control, of the contract and its functions. Must be fixed as soon as possible.

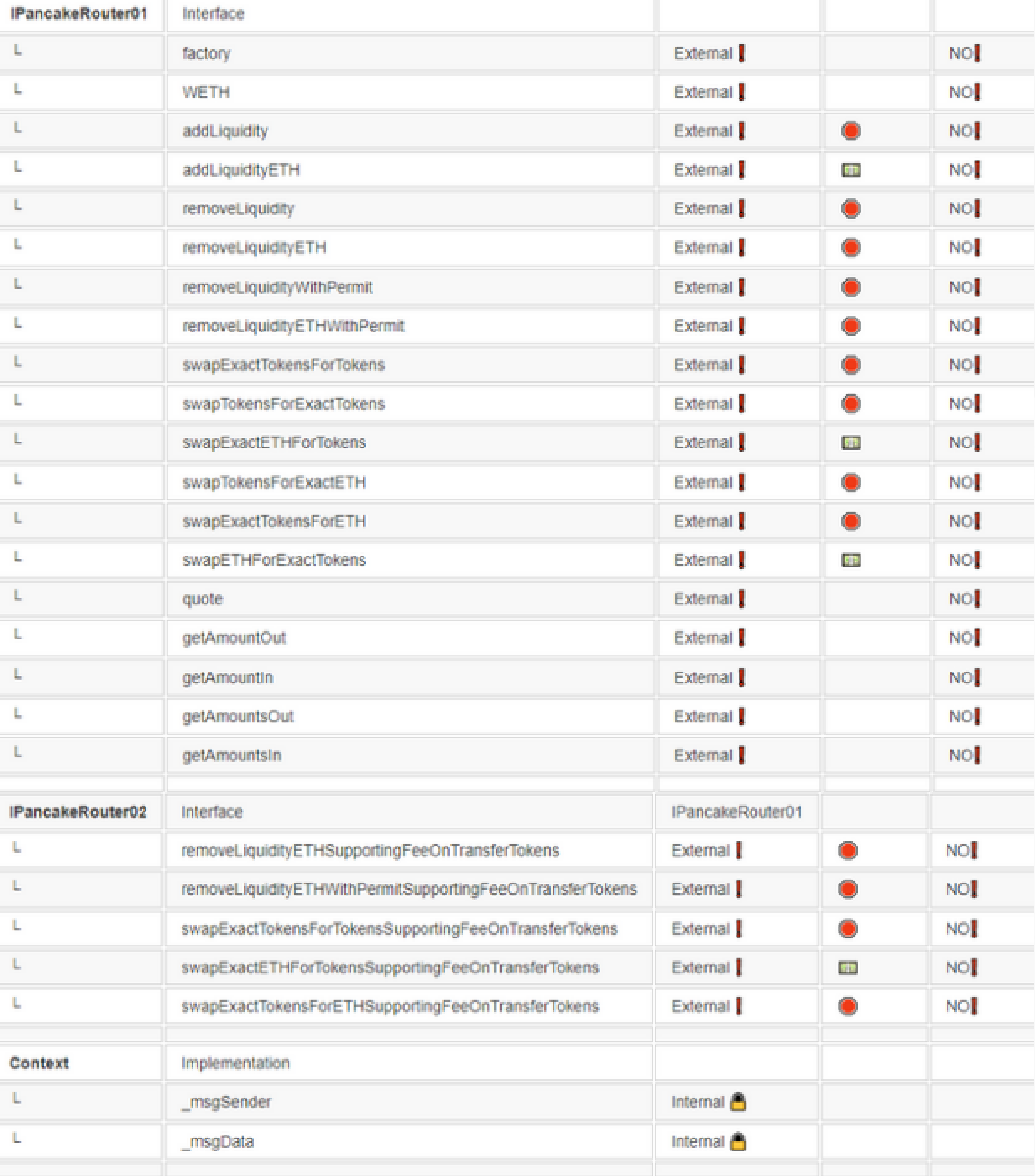
Medium: Bugs or issues with that may be subject to exploit, though their impact is somewhat limited. Must be fxed as soon as possible.

Low: Effects are minimal in isolation and do not pose a signifcant danger to the project or its users. Issues under this classifcation are recommended to be fixed nonetheless.

Automated Analysis

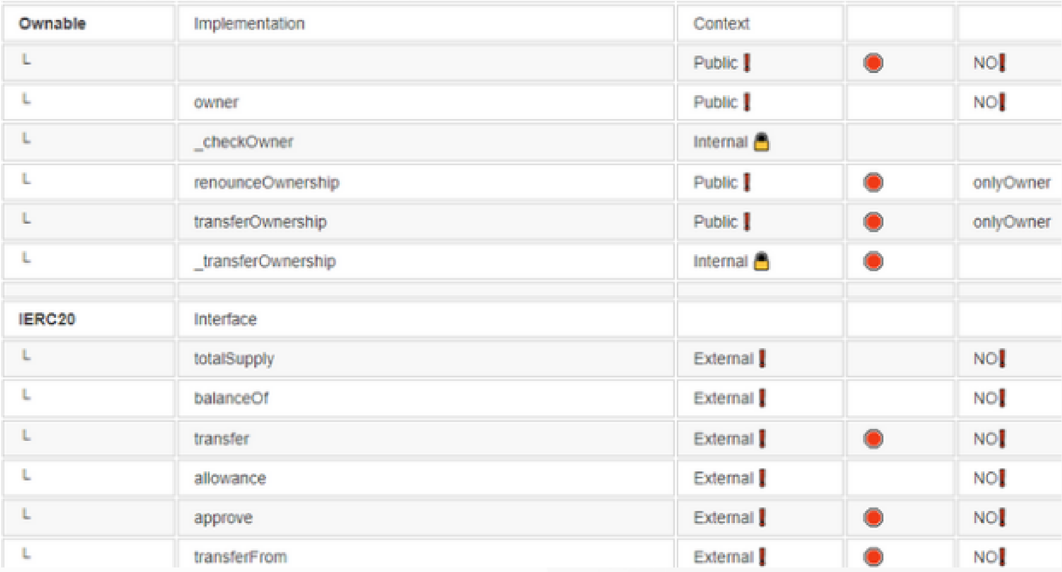


Symbol	Meaning
🔴	Function can modify state
💵	Function is payable

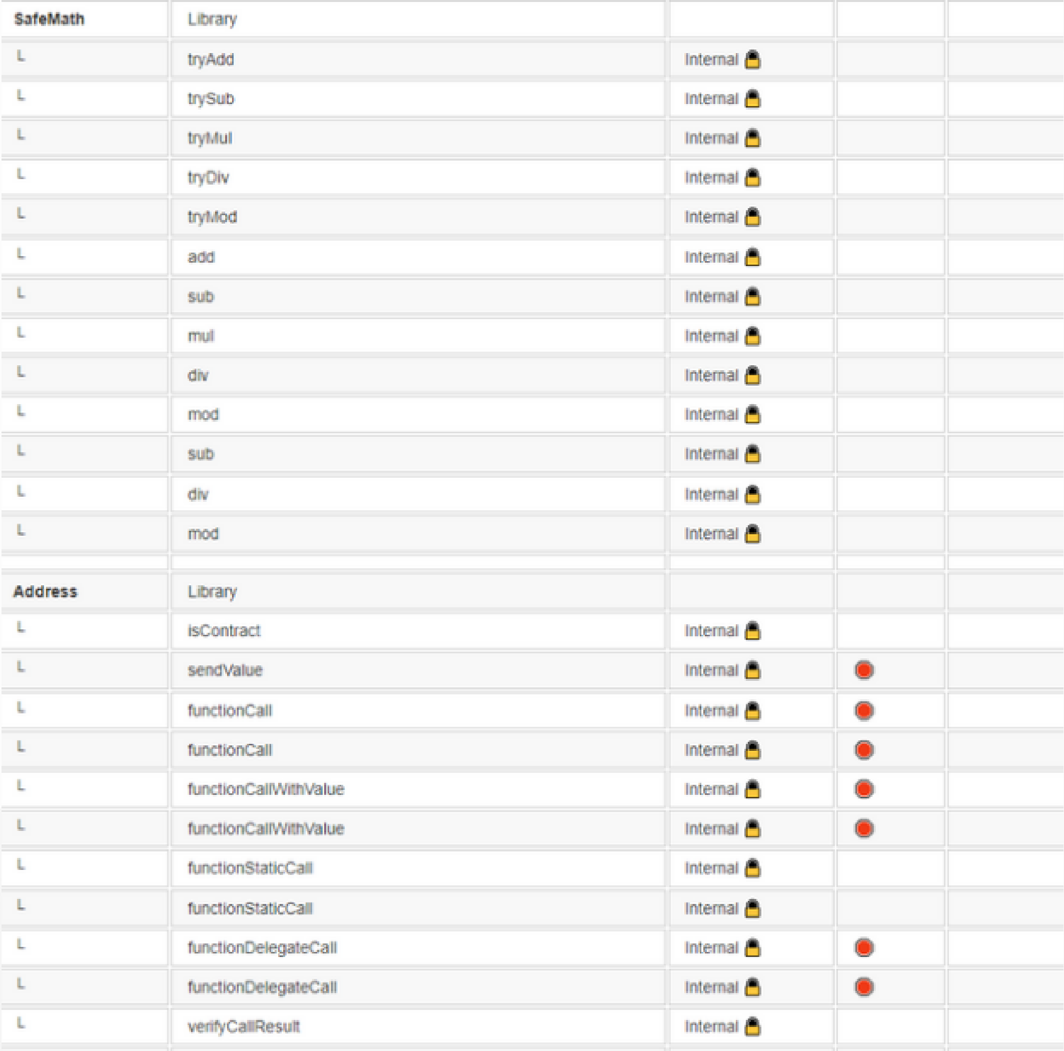


IPancakeRouter01	Interface			
└	factory	External ❗		NO❗
└	WETH	External ❗		NO❗
└	addLiquidity	External ❗	🔴	NO❗
└	addLiquidityETH	External ❗	💵	NO❗
└	removeLiquidity	External ❗	🔴	NO❗
└	removeLiquidityETH	External ❗	🔴	NO❗
└	removeLiquidityWithPermit	External ❗	🔴	NO❗
└	removeLiquidityETHWithPermit	External ❗	🔴	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❗
IPancakeRouter02	Interface	IPancakeRouter01		
└	removeLiquidityETHSupportingFeeOnTransferTokens	External ❗	🔴	NO❗
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🔴	NO❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🔴	NO❗
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🔴	NO❗
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		

Automated Analysis

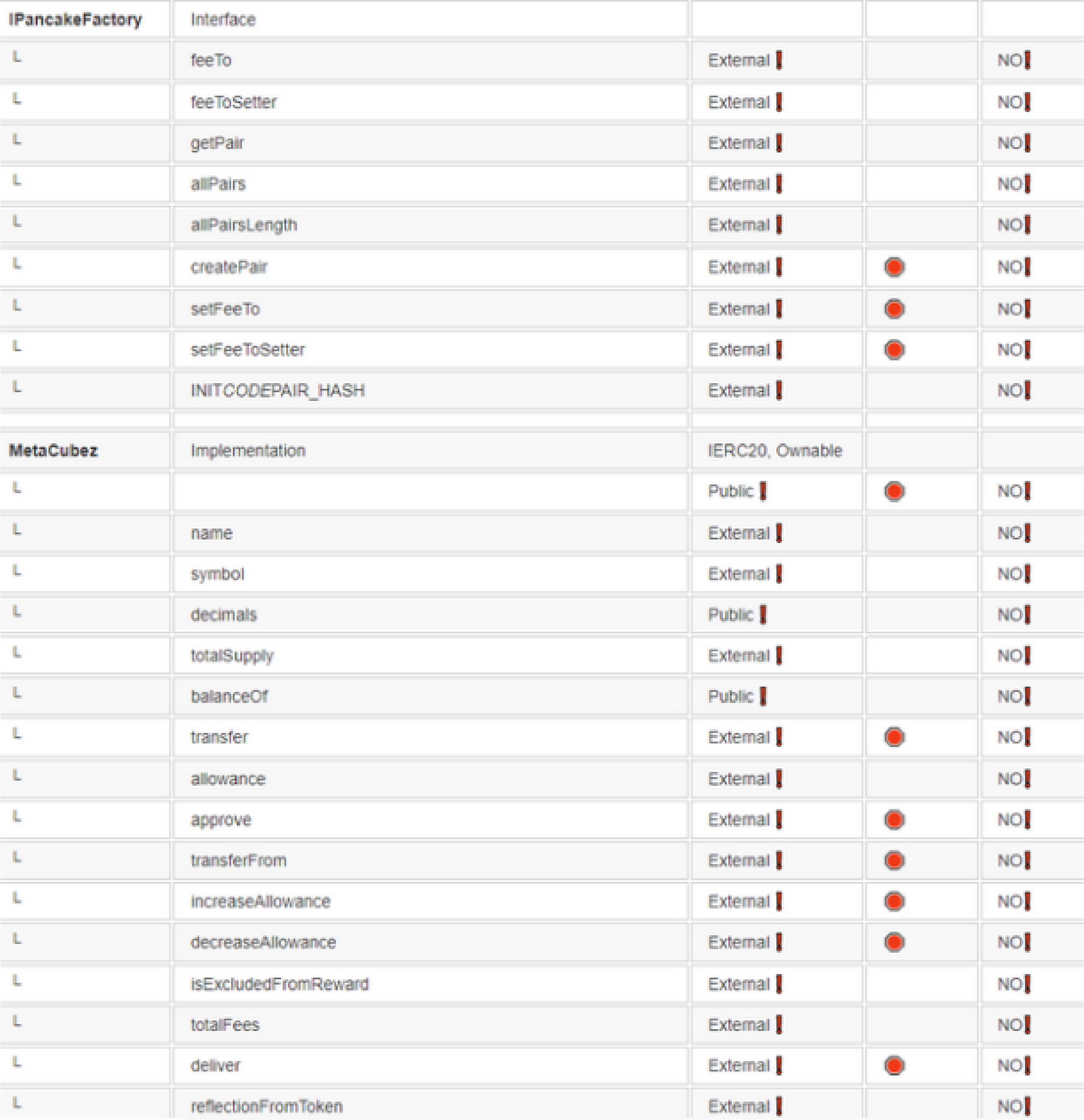


Ownable	Implementation	Context		
└		Public ❗	🔴	NO❗
└	owner	Public ❗		NO❗
└	_checkOwner	Internal 🔒		
└	renounceOwnership	Public ❗	🔴	onlyOwner
└	transferOwnership	Public ❗	🔴	onlyOwner
└	_transferOwnership	Internal 🔒	🔴	
IERC20	Interface			
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🔴	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🔴	NO❗
└	transferFrom	External ❗	🔴	NO❗



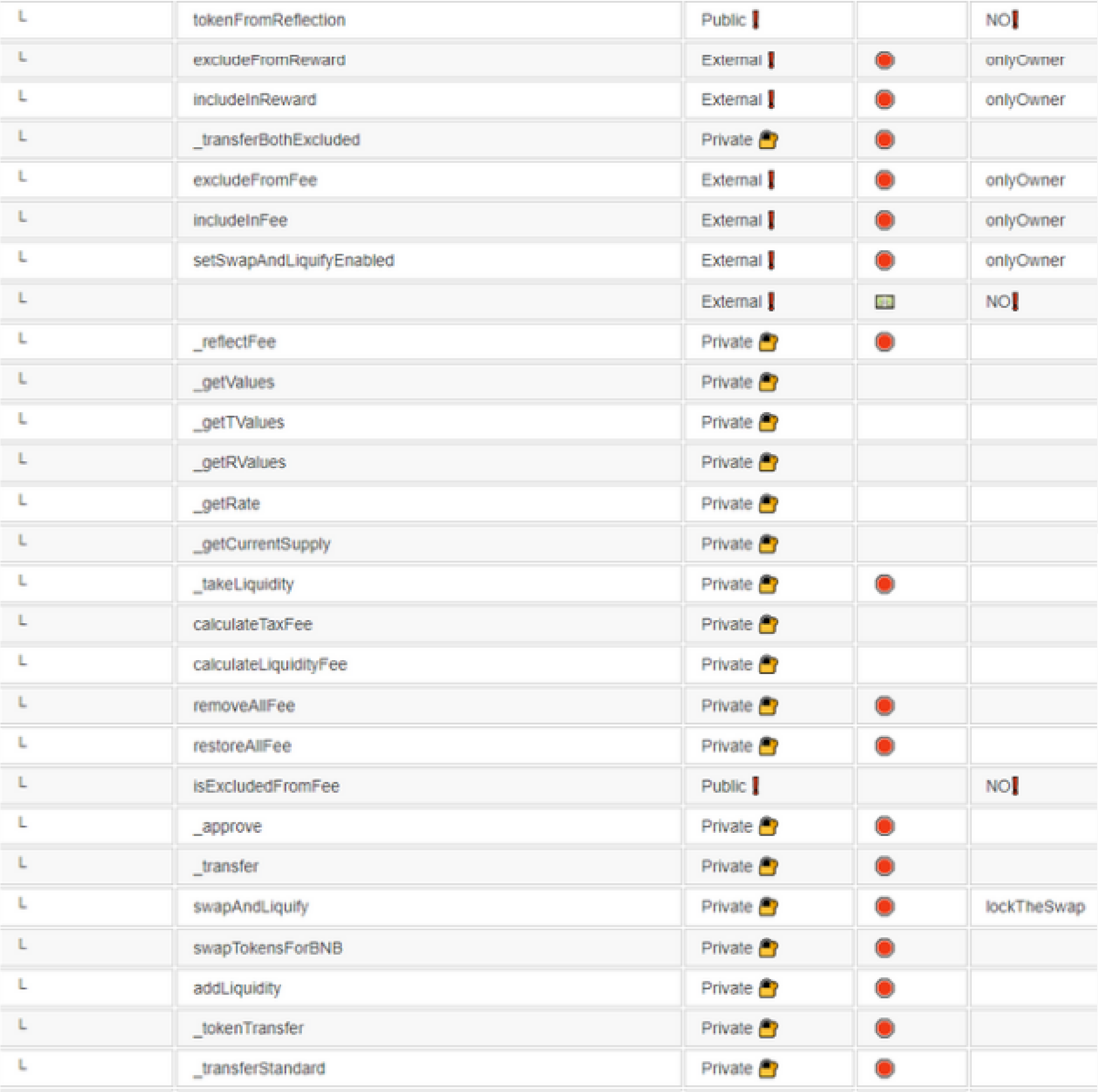
SafeMath	Library			
└	tryAdd	Internal 🔒		
└	trySub	Internal 🔒		
└	tryMul	Internal 🔒		
└	tryDiv	Internal 🔒		
└	tryMod	Internal 🔒		
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	sub	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
Address	Library			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🔴	
└	functionCall	Internal 🔒	🔴	
└	functionCall	Internal 🔒	🔴	
└	functionCallWithValue	Internal 🔒	🔴	
└	functionCallWithValue	Internal 🔒	🔴	
└	functionStaticCall	Internal 🔒		
└	functionStaticCall	Internal 🔒		
└	functionDelegateCall	Internal 🔒	🔴	
└	functionDelegateCall	Internal 🔒	🔴	
└	verifyCallResult	Internal 🔒		

Automated Analysis



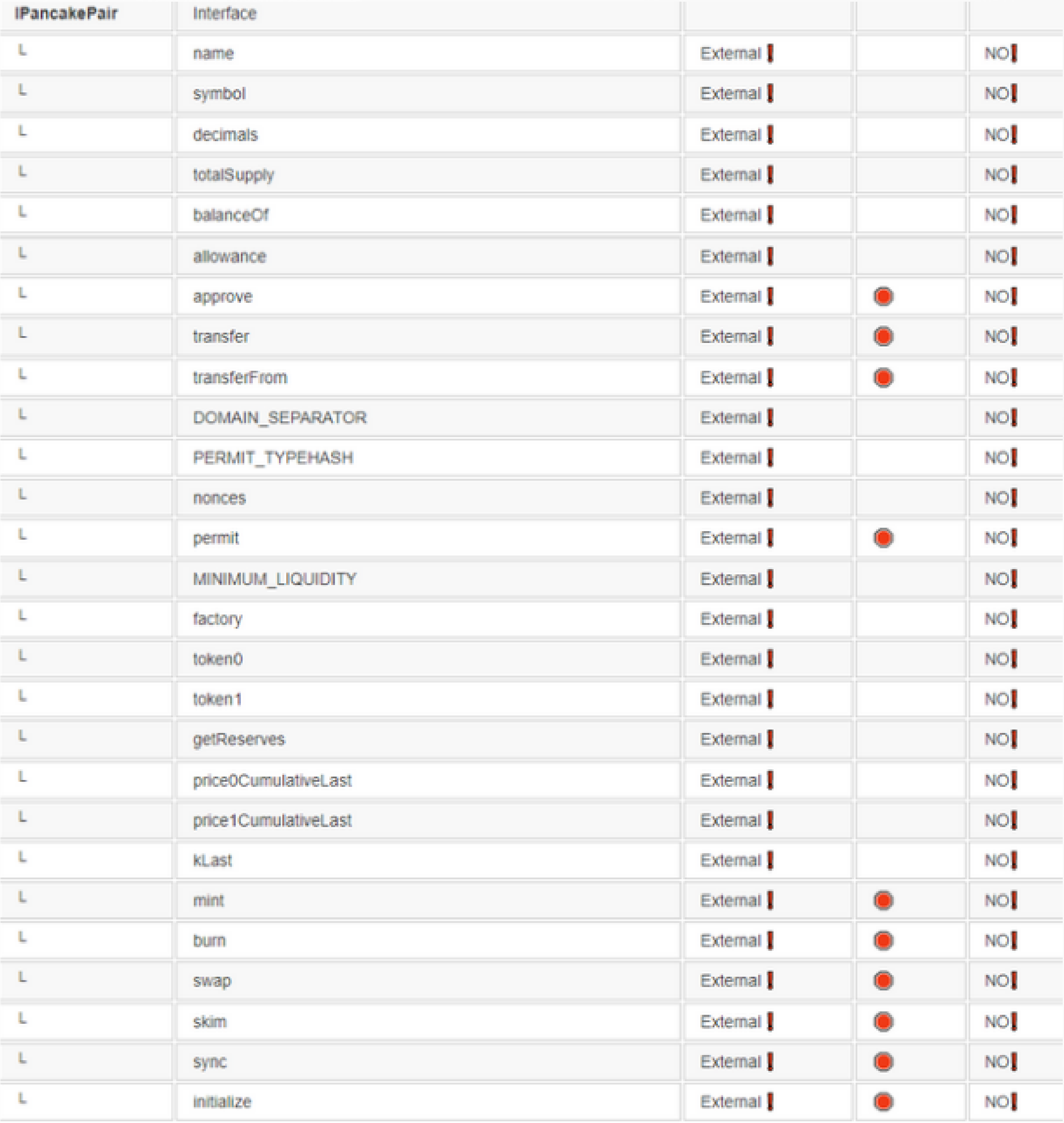
IPancakeFactory	Interface			
└	feeTo	External ❗		NO❗
└	feeToSetter	External ❗		NO❗
└	getPair	External ❗		NO❗
└	allPairs	External ❗		NO❗
└	allPairsLength	External ❗		NO❗
└	createPair	External ❗	●	NO❗
└	setFeeTo	External ❗	●	NO❗
└	setFeeToSetter	External ❗	●	NO❗
└	INIT*CODE*PAIR_HASH	External ❗		NO❗
MetaCubez	Implementation	IERC20, Ownable		
└		Public ❗	●	NO❗
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	decimals	Public ❗		NO❗
└	totalSupply	External ❗		NO❗
└	balanceOf	Public ❗		NO❗
└	transfer	External ❗	●	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	●	NO❗
└	transferFrom	External ❗	●	NO❗
└	increaseAllowance	External ❗	●	NO❗
└	decreaseAllowance	External ❗	●	NO❗
└	isExcludedFromReward	External ❗		NO❗
└	totalFees	External ❗		NO❗
└	deliver	External ❗	●	NO❗
└	reflectionFromToken	External ❗		NO❗

Automated Analysis



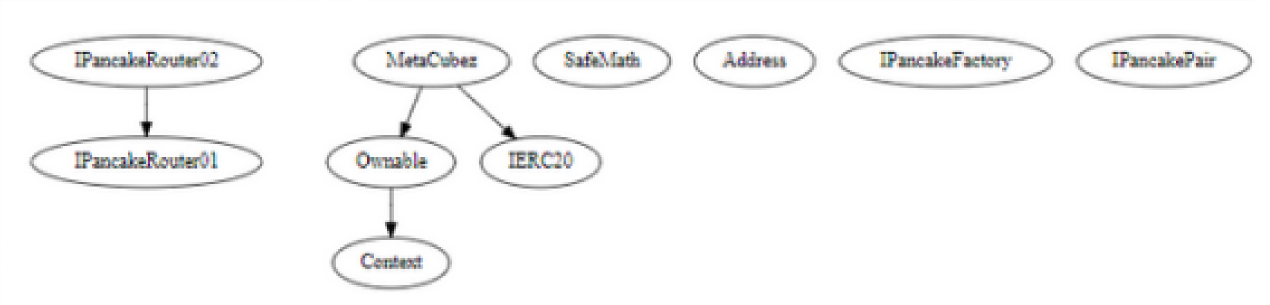
└	tokenFromReflection	Public ❗		NO❗
└	excludeFromReward	External ❗	🛑	onlyOwner
└	includeInReward	External ❗	🛑	onlyOwner
└	_transferBothExcluded	Private 🔐	🛑	
└	excludeFromFee	External ❗	🛑	onlyOwner
└	includeInFee	External ❗	🛑	onlyOwner
└	setSwapAndLiquifyEnabled	External ❗	🛑	onlyOwner
└		External ❗	💵	NO❗
└	_reflectFee	Private 🔐	🛑	
└	_getValues	Private 🔐		
└	_getTValues	Private 🔐		
└	_getRValues	Private 🔐		
└	_getRate	Private 🔐		
└	_getCurrentSupply	Private 🔐		
└	_takeLiquidity	Private 🔐	🛑	
└	calculateTaxFee	Private 🔐		
└	calculateLiquidityFee	Private 🔐		
└	removeAllFee	Private 🔐	🛑	
└	restoreAllFee	Private 🔐	🛑	
└	isExcludedFromFee	Public ❗		NO❗
└	_approve	Private 🔐	🛑	
└	_transfer	Private 🔐	🛑	
└	swapAndLiquify	Private 🔐	🛑	lockTheSwap
└	swapTokensForBNB	Private 🔐	🛑	
└	addLiquidity	Private 🔐	🛑	
└	_tokenTransfer	Private 🔐	🛑	
└	_transferStandard	Private 🔐	🛑	

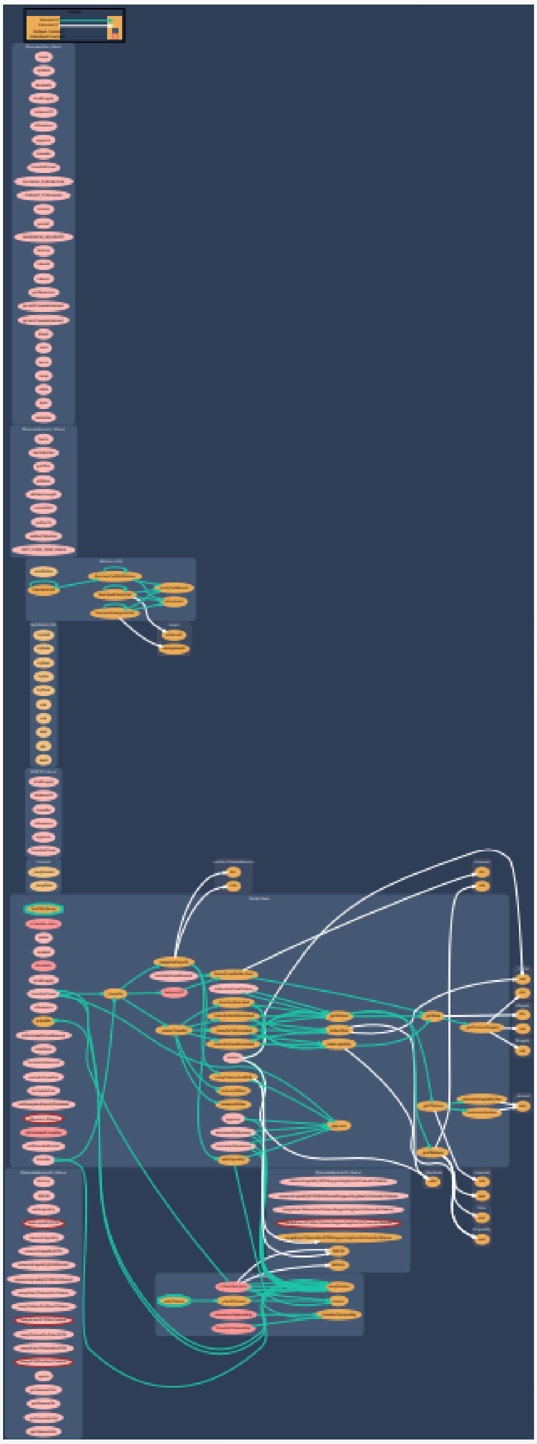
Automated Analysis

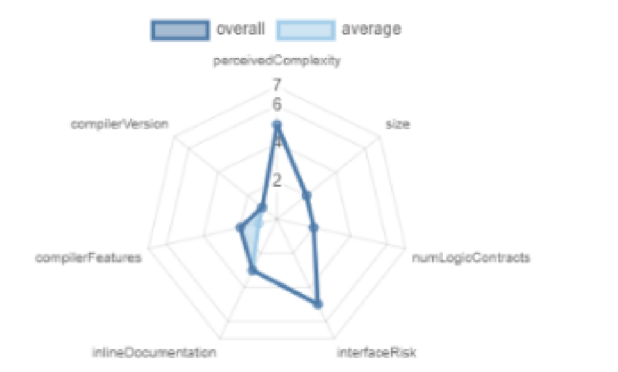


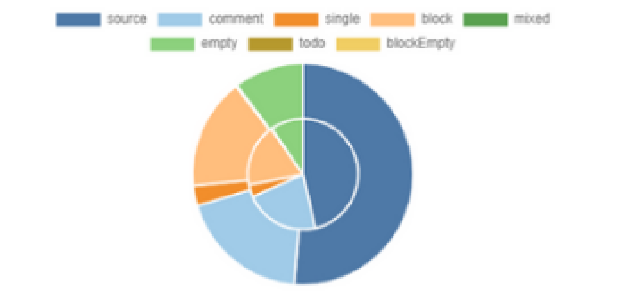
IPancakePair	Interface			
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	decimals	External ❗		NO❗
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🔴	NO❗
└	transfer	External ❗	🔴	NO❗
└	transferFrom	External ❗	🔴	NO❗
└	DOMAIN_SEPARATOR	External ❗		NO❗
└	PERMIT_TYPEHASH	External ❗		NO❗
└	nonces	External ❗		NO❗
└	permit	External ❗	🔴	NO❗
└	MINIMUM_LIQUIDITY	External ❗		NO❗
└	factory	External ❗		NO❗
└	token0	External ❗		NO❗
└	token1	External ❗		NO❗
└	getReserves	External ❗		NO❗
└	price0CumulativeLast	External ❗		NO❗
└	price1CumulativeLast	External ❗		NO❗
└	kLast	External ❗		NO❗
└	mint	External ❗	🔴	NO❗
└	burn	External ❗	🔴	NO❗
└	swap	External ❗	🔴	NO❗
└	skim	External ❗	🔴	NO❗
└	sync	External ❗	🔴	NO❗
└	initialize	External ❗	🔴	NO❗

Inheritance Graph

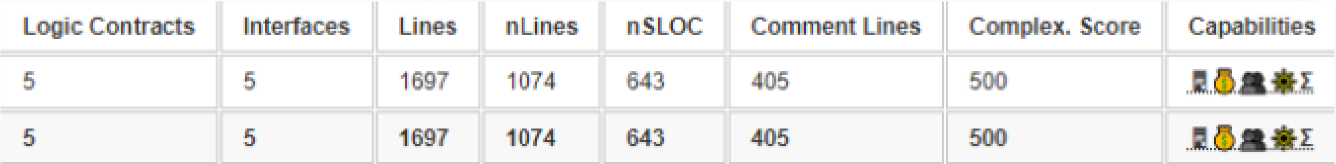




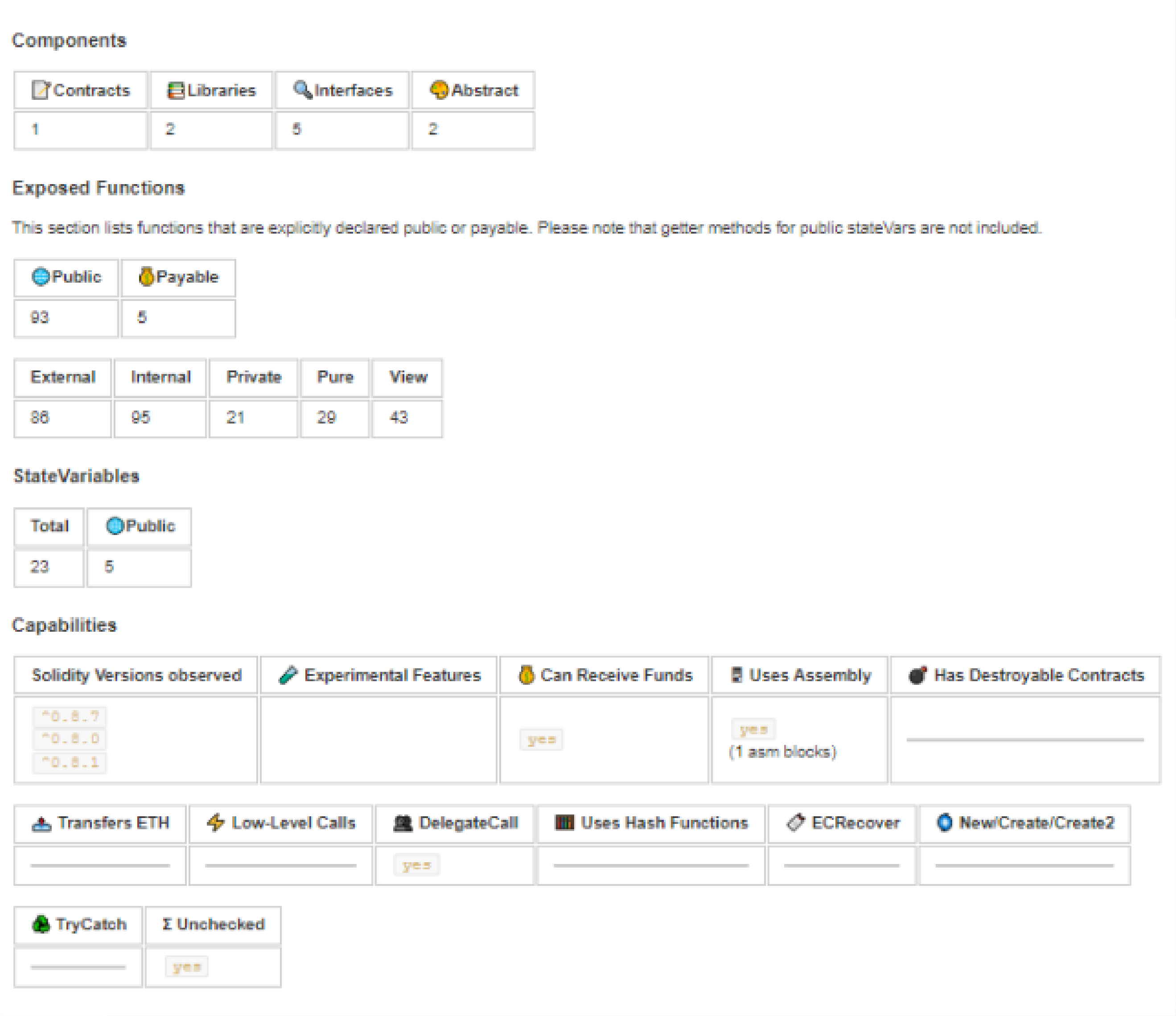




Contract Summary



Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
5	5	1697	1074	643	405	500	🖥💰👥🌀Σ
5	**5**	**1697**	**1074**	**643**	**405**	**500**	🖥💰👥🌀Σ



Components

📝Contracts	📚Libraries	🔍Interfaces	🎨Abstract
1	2	5	2

Exposed Functions

This section lists functions that are explicitly declared public or payable. Please note that getter methods for public stateVars are not included.

🌐Public	💰Payable
93	5

External	Internal	Private	Pure	View
86	95	21	29	43

StateVariables

Total	🌐Public
23	5

Capabilities

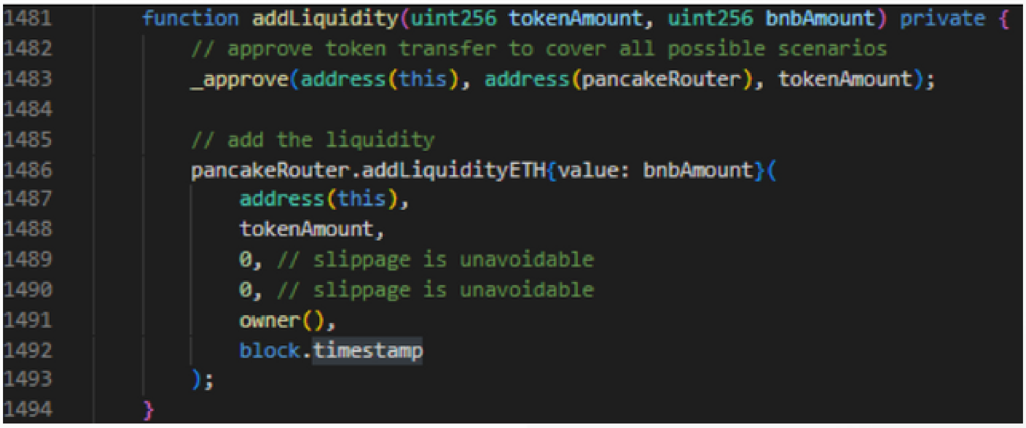
Solidity Versions observed	🧪 Experimental Features	💰 Can Receive Funds	🖥 Uses Assembly	💣 Has Destroyable Contracts
^0.8.7 ^0.8.0 ^0.8.1		yes	yes (1 asm blocks)	—

📤 Transfers ETH	⚡ Low-Level Calls	👥 DelegateCall	🧮 Uses Hash Functions	🔖 ECRecover	🌀 New/Create/Create2
—	—	yes	—	—	—

♻️ TryCatch	Σ Unchecked
—	yes

Manual Review

Auto liquidity is going to an externally owned account

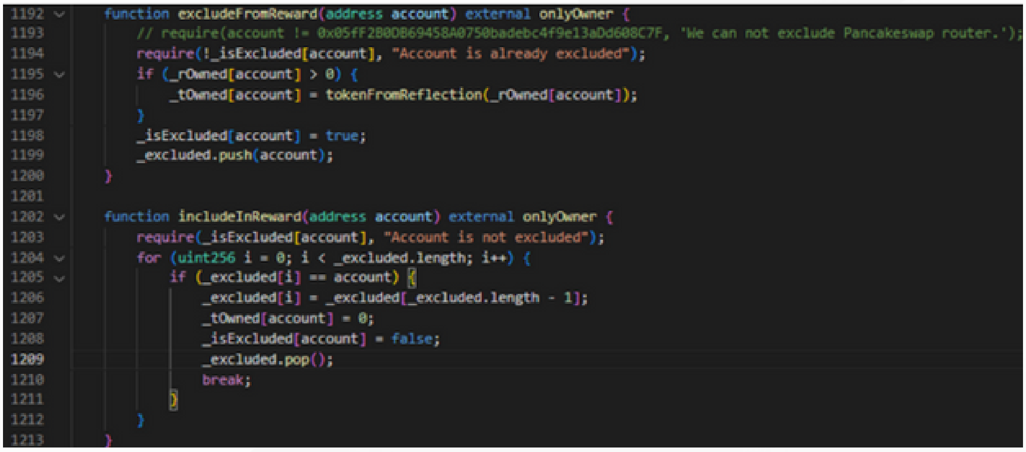


```
    function addLiquidity(uint256 tokenAmount, uint256 bnbAmount) private {
        // approve token transfer to cover all possible scenarios
        _approve(address(this), address(pancakeRouter), tokenAmount);

        // add the liquidity
        pancakeRouter.addLiquidityETH{value: bnbAmount}(
            address(this),
            tokenAmount,
            0, // slippage is unavoidable
            0, // slippage is unavoidable
            owner(),
            block.timestamp
        );
    }
```

Recommendation

Authorizing privileged roles to externally-owned-account (EOA) is dangerous.
Send LP tokens to dead address or unreachable address.

The owner can exclude accounts from rewards



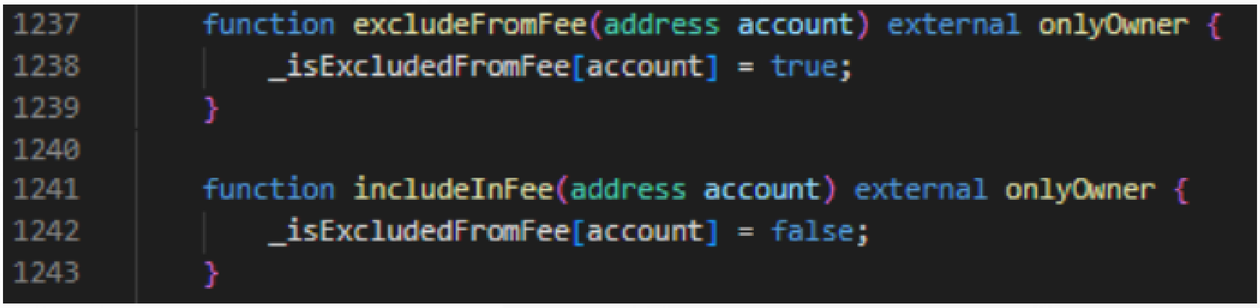
```
    function excludeFromReward(address account) external onlyOwner {
        // require(account != 0x05fF2B0DB69458A0750badebc4f9e13aDd608C7F, 'We can not exclude Pancakeswap router.');
        require(!_isExcluded[account], "Account is already excluded");
        if (_rOwned[account] > 0) {
            _tOwned[account] = tokenFromReflection(_rOwned[account]);
        }
        _isExcluded[account] = true;
        _excluded.push(account);
    }

    function includeInReward(address account) external onlyOwner {
        require(_isExcluded[account], "Account is not excluded");
        for (uint256 i = 0; i < _excluded.length; i++) {
            if (_excluded[i] == account) {
                _excluded[i] = _excluded[_excluded.length - 1];
                _tOwned[account] = 0;
                _isExcluded[account] = false;
                _excluded.pop();
                break;
            }
        }
    }
```

Recommendation

Authorizing privileged roles to exclude accounts from rewards. These cause affect decentralization

Manual Review

The owner can exclude accounts from fees

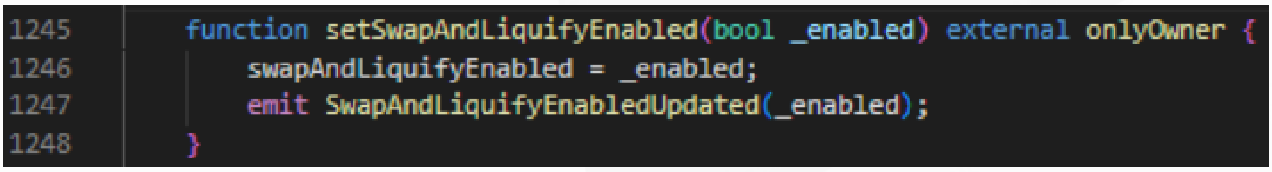


```
    function excludeFromFee(address account) external onlyOwner {
        _isExcludedFromFee[account] = true;
    }

    function includeInFee(address account) external onlyOwner {
        _isExcludedFromFee[account] = false;
    }
```

Recommendation

Authorizing privileged roles to exclude accounts from fees. These cause affect decentralization

The owner can change swap settings



```
    function setSwapAndLiquifyEnabled(bool _enabled) external onlyOwner {
        swapAndLiquifyEnabled = _enabled;
        emit SwapAndLiquifyEnabledUpdated(_enabled);
    }
```

Recommendation

Authorizing privileged roles to enable or disable the swap. These cause affect decentralization

AUDIT REPORT

SecureWise

SMART CONTRACT AUDIT

https://github.com/securewise
https://t.me/securewise
https://securewise.info/

SecureWise