SecureWise

AUDIT REPORT

SecureWise

SMART CONTRACT AUDIT

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: Central Bank of Memes (**PRINT**)

Methodology: Automated Analysis, Manual Code Review

Language: Solidity

Contract Address: 0xb634bAbe1698C8E102416B0f7Fe6A68053dCcb3B

ContractLink: https://etherscan.io/address/0xb634bAbe1698C8E102416B0f7Fe6A68053dCcb3B

Network: ETH

Decimals: 18

Supply: 1,000,000,000

Website: https://www.centralbankofmemes.com/

Twitter: https://twitter.com/BankofMemes_

Telegram: https://t.me/entryportalcentralbankofmemes

Report Date: October 1, 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

 The owner can exclude accounts from rewards

 The owner can exclude accounts from fees

 The owner can set fees with limit up to 25%

 The owner can change swap settings

Central Bank of Memes (PRINT) 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

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
Context	Implementation			
└	_msgSender	Internal		
└	_msgData	Internal		
ERC20	Implementation	Context, IERC20, IERC20Metadata		
└		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		
└	_mint	Internal		
└	_burn	Internal		
└	_approve	Internal		

Automated Analysis

└	_beforeTokenTransfer	Internal 🔒	🔴	
└	_afterTokenTransfer	Internal 🔒	🔴	
Ownable	Implementation	Context		
└		Public ❗	🔴	NO❗
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🔴	onlyOwner
└	transferOwnership	Public ❗	🔴	onlyOwner
└	_setOwner	Private 🔐	🔴	
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 🔒		
Clones	Library			
└	clone	Internal 🔒	🔴	
└	cloneDeterministic	Internal 🔒	🔴	
└	predictDeterministicAddress	Internal 🔒		
└	predictDeterministicAddress	Internal 🔒		
Address	Library			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🔴	
└	functionCall	Internal 🔒	🔴	
└	functionCall	Internal 🔒	🔴	
└	functionCallWithValue	Internal 🔒	🔴	
└	functionCallWithValue	Internal 🔒	🔴	

Automated Analysis

└	functionStaticCall	Internal 🔒		
└	functionStaticCall	Internal 🔒		
└	functionDelegateCall	Internal 🔒	🔴	
└	functionDelegateCall	Internal 🔒	🔴	
└	verifyCallResult	Internal 🔒		
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❗
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❗

Automated Analysis

IUniswapV2Router02	Interface	IUniswapV2Router01		
└	removeLiquidityETHSupportingFeeOnTransferTokens	External		NO
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External		NO
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External		NO
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External		NO
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External		NO
IPinkAntiBot	Interface			
└	setTokenOwner	External		NO
└	onPreTransferCheck	External		NO
IERC20Upgradeable	Interface			
└	totalSupply	External		NO
└	balanceOf	External		NO
└	transfer	External		NO
└	allowance	External		NO
└	approve	External		NO
└	transferFrom	External		NO
IERC20MetadataUpgradeable	Interface	IERC20Upgradeable		
└	name	External		NO
└	symbol	External		NO
└	decimals	External		NO
Initializable	Implementation			
ContextUpgradeable	Implementation	Initializable		
└	*_ContextInit*	Internal		initializer
└	*_ContextInit_unchained*	Internal		initializer
└	_msgSender	Internal		
└	_msgData	Internal		
ERC20Upgradeable	Implementation	Initializable, ContextUpgradeable, IERC20Upgradeable, IERC20MetadataUpgradeable		
└	*_ERC20Init*	Internal		initializer
└	*_ERC20Init_unchained*	Internal		initializer
└	name	Public		NO
└	symbol	Public		NO
└	decimals	Public		NO
└	totalSupply	Public		NO
└	balanceOf	Public		NO
└	transfer	Public		NO

Automated Analysis

└	allowance	Public		NO
└	approve	Public	🔴	NO
└	transferFrom	Public	🔴	NO
└	increaseAllowance	Public	🔴	NO
└	decreaseAllowance	Public	🔴	NO
└	_transfer	Internal 🔒	🔴	
└	_mint	Internal 🔒	🔴	
└	_burn	Internal 🔒	🔴	
└	_approve	Internal 🔒	🔴	
└	_beforeTokenTransfer	Internal 🔒	🔴	
└	_afterTokenTransfer	Internal 🔒	🔴	
OwnableUpgradeable	Implementation	Initializable, ContextUpgradeable		
└	_Ownableinit	Internal 🔒	🔴	initializer
└	_Ownableinit_unchained	Internal 🔒	🔴	initializer
└	owner	Public		NO
└	renounceOwnership	Public	🔴	onlyOwner
└	transferOwnership	Public	🔴	onlyOwner
└	_setOwner	Private 🔐	🔴	
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
└	token0	External		NO
└	token1	External		NO
└	getReserves	External		NO

Automated Analysis

└	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
SafeMathInt	Library			
└	mul	Internal		
└	div	Internal		
└	sub	Internal		
└	add	Internal		
└	abs	Internal		
└	toUint256Safe	Internal		
SafeMathUint	Library			
└	toInt256Safe	Internal		
IterableMapping	Library			
└	get	Public		NO
└	getIndexOfKey	Public		NO
└	getKeyAtIndex	Public		NO
└	size	Public		NO
└	set	Public		NO
└	remove	Public		NO
DividendPayingTokenInterface	Interface			
└	dividendOf	External		NO
└	withdrawDividend	External		NO
DividendPayingTokenOptionalInterface	Interface			
└	withdrawableDividendOf	External		NO
└	withdrawnDividendOf	External		NO
└	accumulativeDividendOf	External		NO

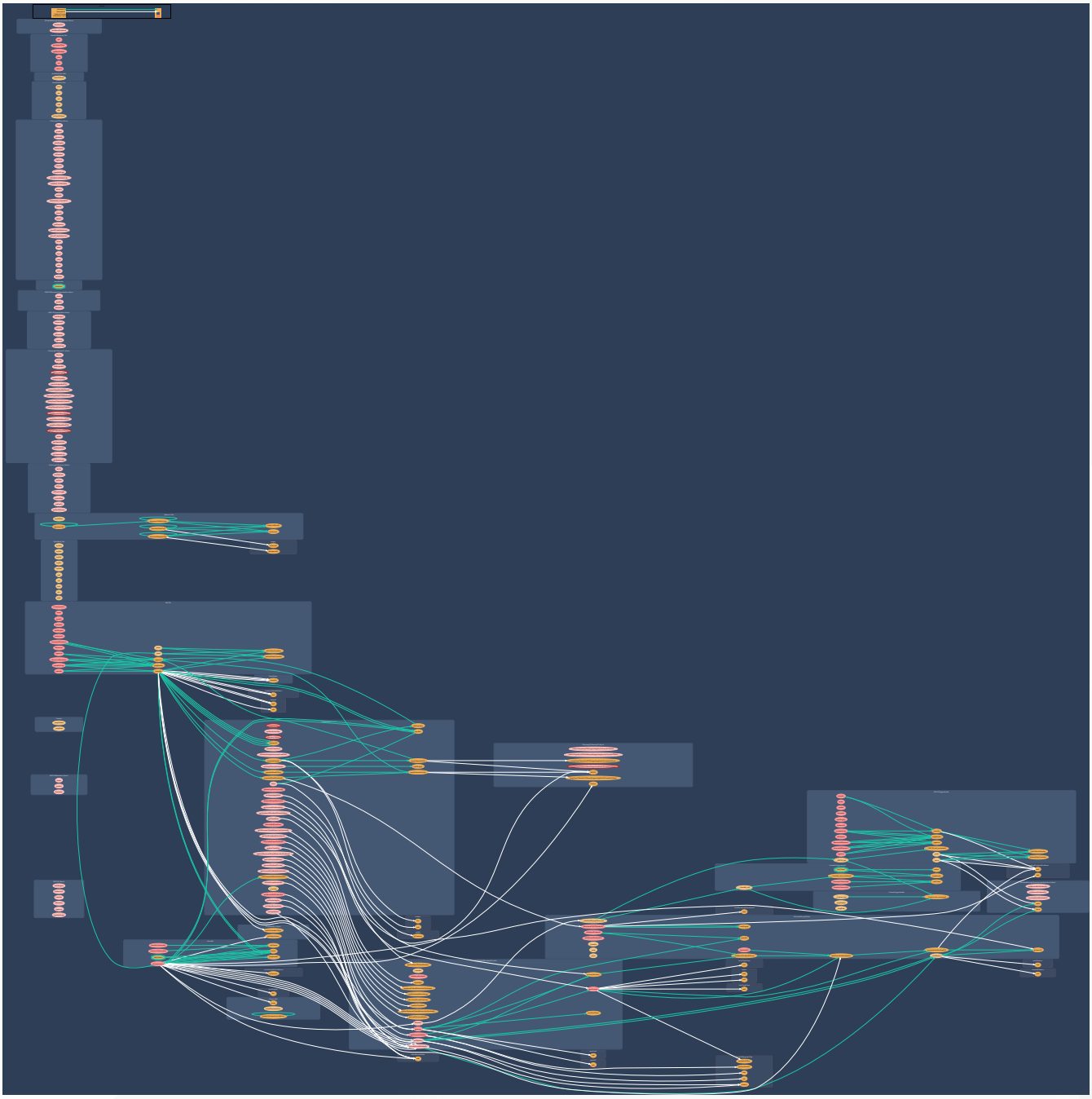
Automated Analysis

DividendPayingToken	Implementation	ERC20Upgradeable, OwnableUpgradeable, DividendPayingTokenInterface, DividendPayingTokenOptionalInterface		
└	_DividendPayingTokeninit	Internal 🔒	🔴	initializer
└	distributeCAKEDividends	Public ❗	🔴	onlyOwner
└	withdrawDividend	Public ❗	🔴	NO❗
└	_withdrawDividendOfUser	Internal 🔒	🔴	
└	dividendOf	Public ❗		NO❗
└	withdrawableDividendOf	Public ❗		NO❗
└	withdrawnDividendOf	Public ❗		NO❗
└	accumulativeDividendOf	Public ❗		NO❗
└	_transfer	Internal 🔒	🔴	
└	_mint	Internal 🔒	🔴	
└	_burn	Internal 🔒	🔴	
└	_setBalance	Internal 🔒	🔴	
BABYTOKENDividendTracker	Implementation	OwnableUpgradeable, DividendPayingToken		
└	initialize	External ❗	🔴	initializer
└	_transfer	Internal 🔒		
└	withdrawDividend	Public ❗		NO❗
└	excludeFromDividends	External ❗	🔴	onlyOwner
└	isExcludedFromDividends	Public ❗		NO❗
└	updateClaimWait	External ❗	🔴	onlyOwner
└	updateMinimumTokenBalanceForDividends	External ❗	🔴	onlyOwner
└	getLastProcessedIndex	External ❗		NO❗
└	getNumberOfTokenHolders	External ❗		NO❗
└	getAccount	Public ❗		NO❗
└	getAccountAtIndex	Public ❗		NO❗
└	canAutoClaim	Private 🔐		
└	setBalance	External ❗	🔴	onlyOwner
└	process	Public ❗	🔴	NO❗
└	processAccount	Public ❗	🔴	onlyOwner
BaseToken	Implementation			
AntiBotBABYTOKEN	Implementation	ERC20, Ownable, BaseToken		
└		Public ❗	💳	ERC20
└	setEnableAntiBot	External ❗	🔴	onlyOwner
└		External ❗	💳	NO❗
└	setSwapTokensAtAmount	External ❗	🔴	onlyOwner
└	excludeFromFees	External ❗	🔴	onlyOwner
└	excludeMultipleAccountsFromFees	External ❗	🔴	onlyOwner
└	setMarketingWallet	External ❗	🔴	onlyOwner

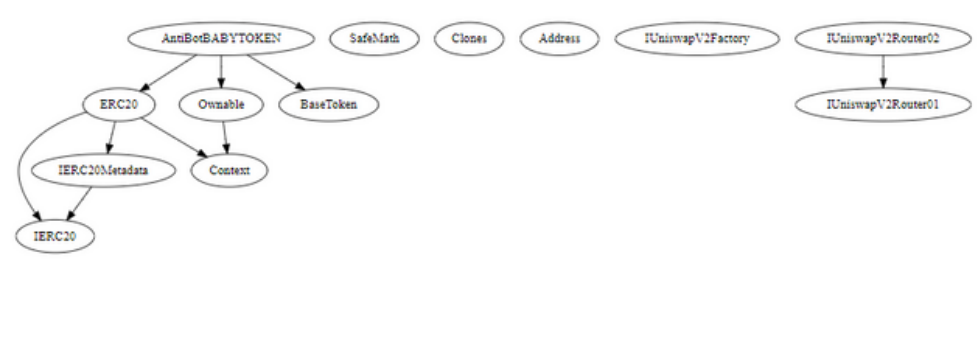
Automated Analysis

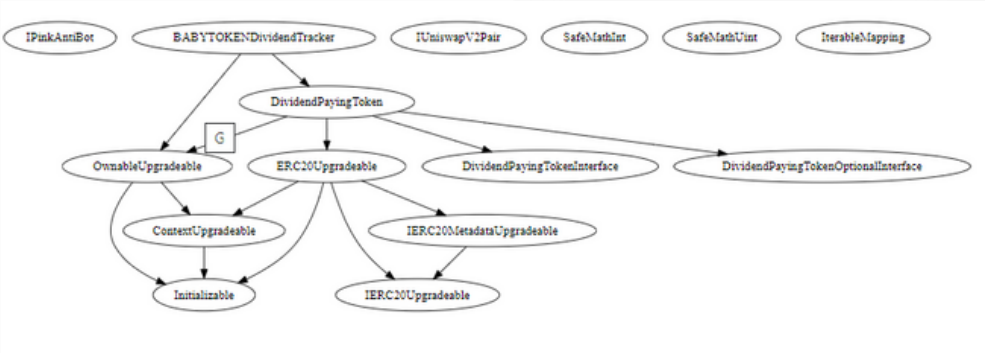
└	setTokenRewardsFee	External		onlyOwner
└	setLiquiditFee	External		onlyOwner
└	setMarketingFee	External		onlyOwner
└	_setAutomatedMarketMakerPair	Private		
└	updateGasForProcessing	Public		onlyOwner
└	updateClaimWait	External		onlyOwner
└	getClaimWait	External		NO
└	updateMinimumTokenBalanceForDividends	External		onlyOwner
└	getMinimumTokenBalanceForDividends	External		NO
└	getTotalDividendsDistributed	External		NO
└	isExcludedFromFees	Public		NO
└	withdrawableDividendOf	Public		NO
└	dividendTokenBalanceOf	Public		NO
└	excludeFromDividends	External		onlyOwner
└	isExcludedFromDividends	Public		NO
└	getAccountDividendsInfo	External		NO
└	getAccountDividendsInfoAtIndex	External		NO
└	processDividendTracker	External		NO
└	claim	External		NO
└	getLastProcessedIndex	External		NO
└	getNumberOfDividendTokenHolders	External		NO
└	_transfer	Internal		
└	swapAndSendToFee	Private		
└	swapAndLiquify	Private		
└	swapTokensForEth	Private		
└	swapTokensForCake	Private		
└	addLiquidity	Private		
└	swapAndSendDividends	Private		

Inheritance Graph

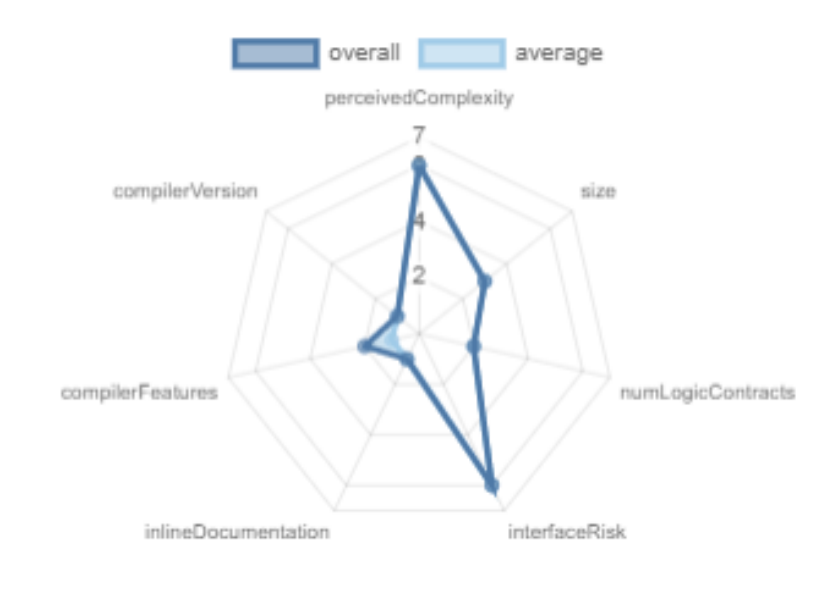


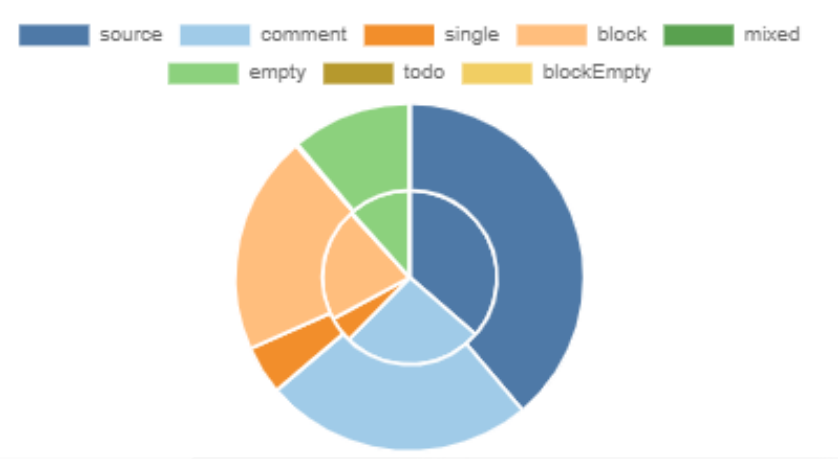
Inheritance Graph





Inheritance Graph





Contract Summary

Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
17	11	3368	2594	1281	1125	1123	
17	**11**	3368	2594	**1281**	**1125**	**1123**	

Components

Contracts	Libraries	Interfaces	Abstract
5	6	11	6

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
163	6

External	Internal	Private	Pure	View
112	184	10	34	82

StateVariables

Total	Public
48	22

Capabilities

Solidity Versions observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
`=0.8.4`		`yes`	`yes` (5 asm blocks)	

Transfers ETH	Low-Level Calls	DelegateCall	Uses Hash Functions	ECRecover	New/Create/Create2
`yes`		`yes`			`yes` → `AssemblyCall:Name:create` → `AssemblyCall:Name:create2`

TryCatch	Σ Unchecked
`yes`	`yes`

Manual Review

The owner can exclude accounts from rewards

```
3037    function excludeFromDividends(address account) external onlyOwner {
3038        dividendTracker.excludeFromDividends(account);
3039    }
```

Recommendation

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

The owner can exclude accounts from fees

```
2915    function excludeFromFees(address account) external onlyOwner {
2916        require(
2917            !_isExcludedFromFees[account],
2918            "BABYTOKEN: Account is already excluded"
2919        );
2920        _isExcludedFromFees[account] = true;
2921
2922        emit ExcludeFromFees(account);
2923    }
```

Recommendation

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

Manual Review

The owner can set fees with limit up to 25%

```
2945    function setTokenRewardsFee(uint256 value) external onlyOwner {
2946        tokenRewardsFee = value;
2947        totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee);
2948        require(totalFees <= 25, "Total fee is over 25%");
2949    }
2950
2951    function setLiquiditFee(uint256 value) external onlyOwner {
2952        liquidityFee = value;
2953        totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee);
2954        require(totalFees <= 25, "Total fee is over 25%");
2955    }
2956
2957    function setMarketingFee(uint256 value) external onlyOwner {
2958        marketingFee = value;
2959        totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee);
2960        require(totalFees <= 25, "Total fee is over 25%");
2961    }
```

Recommendation

-

The owner can change swap settings

```
2907    function setSwapTokensAtAmount(uint256 amount) external onlyOwner {
2908        require(
2909            amount > totalSupply() / 10**5,
2910            "BABYTOKEN: Amount must be greater than 0.001% of total supply"
2911        );
2912        swapTokensAtAmount = amount;
2913    }
```

Recommendation

Authorizing privileged roles to enable or disable the swap.

AUDIT REPORT

SecureWise

SMART CONTRACT AUDIT

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

SecureWise