Cyberscope

Audit Report

PancakeFactory

February 2023

SHA256	2e6835b765ef88e4baa2cbe362e24fffe6cbee24e78f28f7774860605883cd2a
Audited by	© cyberscope

Table of Contents

Review

Audit Updates

Initial Audit	06 Mar 2023

Source Files

Filename	SHA256
interfaces/IERC20.sol	9c75cbedd4aa49570bfc4ca4a8da250ad b1e1e6158ad2c2c5a230ce218adc033
interfaces/IPancakeCallee.sol	a95cc49d2a108030491f500dcfaa196926 a28915ee8ec3bce7ddc2a823e033ec
interfaces/IPancakeERC20.sol	92647340818c895d5b716b97cf6a02269 4347309ea5934787a398e104ed1d441
interfaces/IPancakeFactory.sol	18ff5ffb0e39fca37091ed77356e964fb42 dc6b1f699f6190eaa797dd7b7a23c
interfaces/IPancakePair.sol	3411df2a3f50c805a90e84ed978a65bbce 73a06938f174fc65670dd0628d6534
libraries/Math.sol	68728e7cd44650b0f823189d89d1febec 1b099982dac3edfa6b5745d08d4750e
libraries/SafeMath.sol	7d1ba5983aed2d4b7598fd04c07e22972 9b4d5f543b657c5589d3f3bf796baa2
libraries/UQ112x112.sol	b1595a03b3f9f00282b14f3967b26f6463 c8e4a40fea1b97c725f222aefffc9e
PancakeERC20.sol	7e1bcb1a3b23f12de5bcb3fdb44da220c 36f84872e2f6eee1e0dae6f0aef366a
PancakeFactory.sol	e7c901cbd4d6b1f82f386cdb8546da6d6 3e2f940b79dcfa189fdf8574c2f7b04
PancakePair.sol	e3f46d72297470f36ca18413a9f3062a3d 15a180c79475eb54bf502c653dfcd3

Introduction

This audit is focused on the PancakeFactory contract. The PancakeFactory contract is forked from Pancake Swap. It implements the same functionality as the PancakeFactory contract.

PancakeFactory

The PancakeFactory contract is responsible to create new liquidity pools and trading pairs on the PancakeSwap DEX.

Roles

The PancakeFactory contract does not have Roles.

Diagnostics

● Critical ● Medium ● Minor / Informative

Severity	Code	Description	Status
●	L01	Public Function could be Declared External	Unresolved
●	L18	Multiple Pragma Directives	Unresolved
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L16	Validate Variable Setters	Unresolved
●	L17	Usage of Solidity Assembly	Unresolved

L01 - Public Function could be Declared External

Criticality	Minor / Informative
Location	PancakeRouter.sol#L479,489
Status	Unresolved

Description

A public function is a function that can be called from external contracts or from within the contract itself. An external function is a function that can only be called from external contracts, and cannot be called from within the contract itself.

It's generally a good idea to declare functions as external if they are only intended to be called from external contracts, as this can help to make the contract's code easier to understand and maintain. Declaring a function as external can also help to improve the contract's performance and gas consumption.

```
function getAmountsOut(uint256 amountIn, address[] memory path)
        public
        view
        virtual
        override
        returns (uint256[] memory amounts)
    {
        return PancakeLibrary.getAmountsOut(factory, amountIn, path);
    }

...
```

Recommendation

It's important to choose the appropriate visibility for each function based on how it is intended to be used. Declaring a function as external when it should be public, or vice versa can lead to unnecessary gas consumption.

L18 - Multiple Pragma Directives

Criticality	Minor / Informative
Location	PancakePair.sol#L2 PancakeFactory.sol#L2 PancakeERC20.sol#L2 libraries/UQ112x112.sol#L2 libraries/SafeMath.sol#L2 libraries/Math.sol#L2 interfaces/IPancakePair.sol#L2 interfaces/IPancakeFactory.sol#L2 interfaces/IPancakeERC20.sol#L2 interfaces/IPancakeCallee.sol#L2 interfaces/IERC20.sol#L2
Status	Unresolved

Description

If the contract includes multiple conflicting pragma directives, it may produce unexpected errors. To avoid this, it's important to include the correct pragma directive at the top of the contract and to ensure that it is the only pragma directive included in the contract.

```
pragma solidity =0.5.16;
pragma solidity >=0.5.0;
pragma solidity >=0.5.0 <0.7.0;
```

Recommendation

It is important to include only one pragma directive at the top of the contract and to ensure that it accurately reflects the version of Solidity that the contract is written in.

By including all required compiler options and flags in a single pragma directive, the potential conflicts could be avoided and ensure that the contract can be compiled correctly.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	PancakeFactory.sol#L43,48 interfaces/IPancakeFactory.sol#L23
Status	Unresolved

Description

The Solidity style guide is a set of guidelines for writing clean and consistent Solidity code. Adhering to a style guide can help improve the readability and maintainability of the Solidity code, making it easier for others to understand and work with.

The followings are a few key points from the Solidity style guide:

1. Use camelCase for function and variable names, with the first letter in lowercase (e.g., myVariable, updateCounter).
2. Use PascalCase for contract, struct, and enum names, with the first letter in uppercase (e.g., MyContract, UserStruct, ErrorEnum).
3. Use uppercase for constant variables and enums (e.g., MAX_VALUE, ERROR_CODE).
4. Use indentation to improve readability and structure.
5. Use spaces between operators and after commas.
6. Use comments to explain the purpose and behavior of the code.
7. Keep lines short (around 120 characters) to improve readability.

```
address _feeTo
address _feeToSetter
function INIT_CODE_PAIR_HASH() external view returns (bytes32);
```

Recommendation

By following the Solidity naming convention guidelines, the codebase increased the readability, maintainability, and makes it easier to work with.

Find more information on the Solidity documentation
https://docs.soliditylang.org/en/v0.8.17/style-guide.html#naming-convention.

L16 - Validate Variable Setters

Criticality	Minor / Informative
Location	PancakeFactory.sol#L19,45,50
Status	Unresolved

Description

The contract performs operations on variables that have been configured on user-supplied input. These variables are missing of proper check for the case where a value is zero. This can lead to problems when the contract is executed, as certain actions may not be properly handled when the value is zero.

```
feeToSetter = _feeToSetter
feeTo = _feeTo
```

Recommendation

By adding the proper check, the contract will not allow the variables to be configured with zero value. This will ensure that the contract can handle all possible input values and avoid unexpected behavior or errors. Hence, it can help to prevent the contract from being exploited or operating unexpectedly.

L17 - Usage of Solidity Assembly

Criticality	Minor / Informative
Location	PancakeFactory.sol#L33
Status	Unresolved

Description

Using assembly can be useful for optimizing code, but it can also be error-prone. It's important to carefully test and debug assembly code to ensure that it is correct and does not contain any errors.

Some common types of errors that can occur when using assembly in Solidity include Syntax, Type, Out-of-bounds, Stack, and Revert.

```
assembly {
            pair := create2(0, add(bytecode, 32), mload(bytecode),
salt)
        }
```

Recommendation

It is recommended to use assembly sparingly and only when necessary, as it can be difficult to read and understand compared to Solidity code.

Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
IERC20	Interface			
	name	External		-
	symbol	External		-
	decimals	External		-
	totalSupply	External		-
	balanceOf	External		-
	allowance	External		-
	approve	External	✓	-
	transfer	External	✓	-
	transferFrom	External	✓	-
IPancakeCallee	Interface			
	pancakeCall	External	✓	-
IPancakeERC20	Interface			
	name	External		-
	symbol	External		-
	decimals	External		-
	totalSupply	External		-
	balanceOf	External		-
	allowance	External		-
	approve	External	✓	-
	transfer	External	✓	-

	transferFrom	External	✓	-
	DOMAIN_SEPARATOR	External		-
	PERMIT_TYPEHASH	External		-
	nonces	External		-
	permit	External	✓	-
IPancakeFactory	Interface			
	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-
	INIT_CODE_PAIR_HASH	External		-
IPancakeMigrator	Interface			
	migrate	External	✓	-
IPancakePair	Interface			
	name	External		-
	symbol	External		-
	decimals	External		-
	totalSupply	External		-
	balanceOf	External		-
	allowance	External		-
	approve	External	✓	-
	transfer	External	✓	-

	transferFrom	External	✓	-
	DOMAIN_SEPARATOR	External		-
	PERMIT_TYPEHASH	External		-
	nonces	External		-
	permit	External	✓	-
	MINIMUM_LIQUIDITY	External		-
	factory	External		-
	token0	External		-
	token1	External		-
	getReserves	External		-
	price0CumulativeLast	External		-
	price1CumulativeLast	External		-
	kLast	External		-
	mint	External	✓	-
	burn	External	✓	-
	swap	External	✓	-
	skim	External	✓	-
	sync	External	✓	-
	initialize	External	✓	-
IPancakeRouter01	Interface			
	factory	External		-
	WETH	External		-
	addLiquidity	External	✓	-
	addLiquidityETH	External	Payable	-
	removeLiquidity	External	✓	-
	removeLiquidityETH	External	✓	-
	removeLiquidityWithPermit	External	✓	-
	removeLiquidityETHWithPermit	External	✓	-

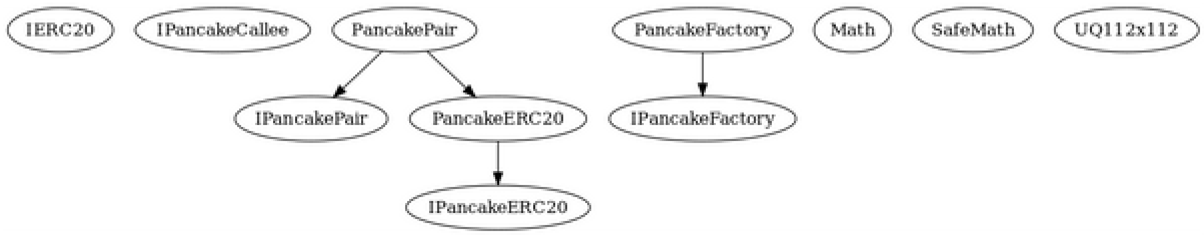
	swapExactTokensForTokens	External	✓	-
	swapTokensForExactTokens	External	✓	-
	swapExactETHForTokens	External	Payable	-
	swapTokensForExactETH	External	✓	-
	swapExactTokensForETH	External	✓	-
	swapETHForExactTokens	External	Payable	-
	quote	External		-
	getAmountOut	External		-
	getAmountIn	External		-
	getAmountsOut	External		-
	getAmountsIn	External		-
IPancakeRouter02	Interface	IPancakeRouter01		
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
IWETH	Interface			
	deposit	External	Payable	-
	transfer	External	✓	-
	withdraw	External	✓	-
Babylonian	Library			
	sqrt	Internal		

Math	Library			
	min	Internal		
	sqrt	Internal		
PancakeLibrary	Library			
	sortTokens	Internal		
	pairFor	Internal		
	getReserves	Internal		
	quote	Internal		
	getAmountOut	Internal		
	getAmountIn	Internal		
	getAmountsOut	Internal		
	getAmountsIn	Internal		
SafeMath	Library			
	add	Internal		
	sub	Internal		
	mul	Internal		
UQ112x112	Library			
	encode	Internal		
	uqdiv	Internal		
WBNB	Implementation			
		Public	Payable	-
	deposit	Public	Payable	-
	withdraw	Public	✓	-
	totalSupply	Public		-
	approve	Public	✓	-

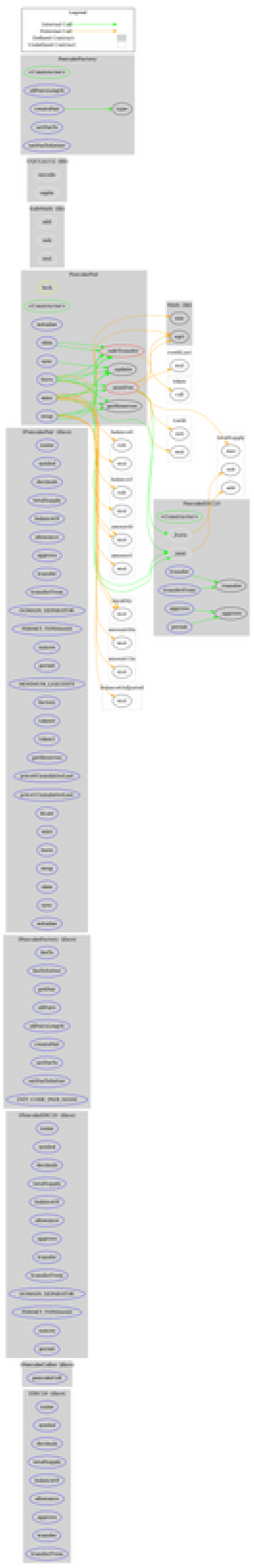
	transfer	Public	✓	-
	transferFrom	Public	✓	-
PancakeRouter	Implementation	IPancakeRouter02		
		Public	✓	-
		External	Payable	-
	_addLiquidity	Internal	✓	
	addLiquidity	External	✓	ensure
	addLiquidityETH	External	Payable	ensure
	removeLiquidity	Public	✓	ensure
	removeLiquidityETH	Public	✓	ensure
	removeLiquidityWithPermit	External	✓	-
	removeLiquidityETHWithPermit	External	✓	-
	removeLiquidityETHSupportingFeeOnTransferTokens	Public	✓	ensure
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	_swap	Internal	✓	
	swapExactTokensForTokens	External	✓	ensure
	swapTokensForExactTokens	External	✓	ensure
	swapExactETHForTokens	External	Payable	ensure
	swapTokensForExactETH	External	✓	ensure
	swapExactTokensForETH	External	✓	ensure
	swapETHForExactTokens	External	Payable	ensure
	_swapSupportingFeeOnTransferTokens	Internal	✓	
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	ensure
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	ensure
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	ensure
	quote	Public		-

	getAmountOut	Public		-
	getAmountIn	Public		-
	getAmountsOut	Public		-
	getAmountsIn	Public		-
PancakeRouter01	Implementation	IPancakeRouter01		
		Public	✓	-
		External	Payable	-
	_addLiquidity	Private	✓	
	addLiquidity	External	✓	ensure
	addLiquidityETH	External	Payable	ensure
	removeLiquidity	Public	✓	ensure
	removeLiquidityETH	Public	✓	ensure
	removeLiquidityWithPermit	External	✓	-
	removeLiquidityETHWithPermit	External	✓	-
	_swap	Private	✓	
	swapExactTokensForTokens	External	✓	ensure
	swapTokensForExactTokens	External	✓	ensure
	swapExactETHForTokens	External	Payable	ensure
	swapTokensForExactETH	External	✓	ensure
	swapExactTokensForETH	External	✓	ensure
	swapETHForExactTokens	External	Payable	ensure
	quote	Public		-
	getAmountOut	Public		-
	getAmountIn	Public		-
	getAmountsOut	Public		-
	getAmountsIn	Public		-

Inheritance Graph



Flow Graph



Summary

PancakeFactory contract implements a utility mechanism. This audit investigates security issues, business logic concerns, and potential improvements.

Disclaimer

The information provided in this report does not constitute investment, financial or trading advice and you should not treat any of the document's content as such. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes nor may copies be delivered to any other person other than the Company without Cyberscope's prior written consent. This report is not nor should be considered an “endorsement” or “disapproval” of any particular project or team. This report is not nor should be regarded as an indication of the economics or value of any “product” or “asset” created by any team or project that contracts Cyberscope to perform a security assessment. This document does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors' business, business model or legal compliance. This report should not be used in any way to make decisions around investment or involvement with any particular project. This report represents an extensive assessment process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk Cyberscope's position is that each company and individual are responsible for their own due diligence and continuous security Cyberscope's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies and in no way claims any guarantee of security or functionality of the technology we agree to analyze. The assessment services provided by Cyberscope are subject to dependencies and are under continuing development. You agree that your access and/or use including but not limited to any services reports and materials will be at your sole risk on an as-is where-is and as-available basis Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives false negatives and other unpredictable results. The services may access and depend upon multiple layers of third parties.

About Cyberscope

Cyberscope is a blockchain cybersecurity company that was founded with the vision to make web3.0 a safer place for investors and developers. Since its launch, it has worked with thousands of projects and is estimated to have secured tens of millions of investors' funds.

Cyberscope is one of the leading smart contract audit firms in the crypto space and has built a high-profile network of clients and partners.

The Cyberscope team

https://www.cyberscope.io