HashEx
BLOCKCHAIN SECURITY

rAsko

smart contracts
final audit report

October 2021

hashex.org

contact@hashex.org

Contents

1. Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the disclaimer below – please make sure to read it in full.

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 HashEx and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers, and other representatives) (HashEx) owe no duty of care towards you or any other person, nor does HashEx 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 HashEx 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, HashEx hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against HashEx, 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 the use of this report, and any reliance on this report. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed. HashEx owns all copyright rights to the text, images, photographs, and other content provided in the following document. When using or sharing partly or in full, third parties must provide a direct link to the original document mentioning the author (hashex.org).

2. Overview

HashEx was commissioned by the rAsko team to perform an audit of their smart contracts. The audit was conducted between September 24 and October 1, 2021.The code located in @MarkStone1232/rAsko-swap-smart-contracts GitHub repository was audited after 8efd0e commit.

The purpose of this audit was to achieve the following:

- Identify potential security issues with smart contracts.
- Formally check the logic behind given smart contracts.

Information in this report should be used for understanding the risk exposure of smart contracts, and as a guide to improving the security posture of smart contracts by remediating the issues that were identified.

Update: rAsko team has responded to this report. The updated code is located in GitHub repository after cc7876 commit.

Update: rAsko team has responded to this report. The updated code is located in GitHub repository after 5dc88a commit.

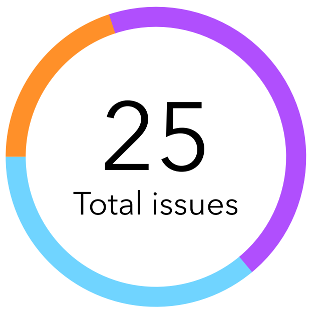
2.1 Summary

Project name	rAsko
URL	https://asko.finance/rasko
Platform	Binance Smart Chain
Language	Solidity

2.2 Contracts

Name	Address
rAskoSwapLibrary	
rAskoFarm	`0xeA65Df6D4a1D597b21CD96a8dAA816D3d0e428e2`
APYCalculator	`0x0f4298Af0d2B5b79cd35c448f7C2e5c75ACd5874`
rAskoSwapRouter	`0xb77621d546c30526BCDdCd52A3B67C4a58F16d59`

3. Found issues



●	High	5 (20%)
●	Medium	11 (44%)
●	Low	9 (36%)

C2f. rAskoSwapLibrary

ID	Severity	Title	Status
C2fI65	● Low	Useless call	✓ Resolved
C2fI66	● Low	Development logging	✓ Resolved

C30. rAskoFarm

ID	Severity	Title	Status
C30I68	● High	Admin's extra powers	✓ Resolved
C30I70	● High	Unsuitable balance checking	✓ Resolved
C30I67	● Medium	Input data is not checked	✓ Resolved
C30I69	● Medium	Input data is not checked	✓ Resolved
C30I6a	● Medium	LP-Token duplication	✓ Resolved

C30I71	● Medium	Insufficient amount of rAsko tokens possibility	✓ Acknowledged
C30I7a	● Medium	Deflationary tokens are not supported	✓ Acknowledged
C30I7d	● Medium	Returned values of transfer are ignored	✓ Acknowledged
C30I6b	● Low	Immutable variable	✓ Resolved
C30I6c	● Low	Unused variable	✓ Resolved
C30I6d	● Low	Implicit visibility modifier	✓ Resolved
C30I6e	● Low	Lack of logging	✓ Partially fixed
C30I6f	● Low	Run out of gas possibility	✓ Acknowledged
C30I79	● Low	rAsko token cannot be in the pools	✓ Acknowledged

C31. APYCalculator

ID	Severity	Title	Status
C31I72	● High	An Admin can't be removed	✓ Resolved
C31I7b	● Medium	Outdated oracle's data	✓ Acknowledged

C32. rAskoSwapRouter

ID	Severity	Title	Status
C32I75	● High	Sending ETH by swap functions	✓ Acknowledged
C32I76	● High	Locking ETH on the contract	✓ Acknowledged
C32I74	● Medium	Transferring ETH by swap functions	✓ Resolved

C32l77	● Medium	Unrestricted fees amount	✓ Acknowledged
C32l78	● Medium	Inappropriate fees values	✓ Acknowledged
C32l7c	● Medium	Returned value of transferFrom is ignored	✓ Acknowledged
C32l73	● Low	No events for admin functions	✓ Acknowledged

4. Contracts

C2f. rAskoSwapLibrary

Overview

Implementation of the UniSwap Library expanding the one concerning rAsko's Router fees.

Issues

C2fI65 Useless call ● Low ✓ Resolved

`getReserves` calls function `pairFor` without saving a result(L52).

Recommendation

The call might be removed since it's called at L54.

Update

The issue was fixed. The pointed function call has been removed.

C2fI66 Development logging ● Low ✓ Resolved

Functions getAmountOut and getAmountIn contain development logging, which is part of Hardhat's tool.

Update

The issue was fixed.

C30. rAskoFarm

Overview

Farming contract. The main contract which a user interacts with.

Issues

C30I68 Admin's extra powers ● High ✓ Resolved

The owner is able to change the `migrator`. That means that they might set a vulnerable implementation on purpose. Such implementation can be used to steal tokens because all funds are approved to the migrator at L167.

Recommendation

If a certain migrator is needed, it should be set once and be immutable.

Update

The was fixed by removing the `migrator` and function `migrate`

C30I70 Unsuitable balance checking ● High ✓ Resolved

In the progress of migration, there is a check that compares LP balances of the old token and the new one, deciding if they're equal. This is not always correct because when a new pair is created, some amount of the new LP tokens is sent to the zero address, and the returned amount of LP tokens won't be the same.

In case the new pair is already created, the farming contract won't ever have the same balance since it's calculated depending on current reserves and totalSupply.

One more problem of the migration is changing underlying tokens rate, which can be exploited.

Recommendation

Since the check can break migration, it's recommended to either not check the balances or to calculate the new balance correctly before the checking.

Update

The issue was fixed. The mentioned balance checking has been removed as well as the function, which it was made from

C30I67 Input data is not checked ● Medium ✓ Resolved

Input parameter of function updateMultiplier is not capped. The owner is able to make `BONUS_MULTIPLIER` too high. That can over-boost the reward amount.

Recommendation

We recommend capping such values.

rAsko response

This isn’t an issue. The `updateMultiplier` method is an admin function and will only be called with complete care. If it is capped we limit ourselves. Furthermore if we make the cap variable it is the same issue just with an extra step.

Update

The issue was fixed at commit 5dc88a0f351d8818690c2867dcfe2f7b129a1409.

C30I69 Input data is not checked ● Medium ✓ Resolved

Input parameter of the method `changeApyCalculator`, which sets a value to the `apyCalculator` variable, is not checked. Setting it to the variable address(0) or just the wrong address breaks the function updatePool.

Recommendation

We recommend checking that the address is not zero at least.

rAsko response

This is also not an issue, changeApyCalculator is an admin function that the utmost care will be taken when calling. Furthermore, calling the method with the 0 address or anything else defaults the value to 0 if it doesn't exist. It will throw an error which is what we intend if there is a misspelt address with the apy calculator so it won’t break it in the sense that everyone's tokens and rewards will still be safe.

Update

The issue was fixed at commit 5dc88a0f351d8818690c2867dcfe2f7b129a1409.

C30I6a LP-Token duplication ● Medium ✓ Resolved

In the `add()` function a token that is already in another pool can be added. In that case, calculations of rewards will be wrong.

Recommendation

It's recommended to check whether the LP-Token has already been added before

Update

The issue was fixed. Function `add()` checks whether an adding LP-Token has already been added.

C30I71 Insufficient amount of rAsko tokens possibility ● Medium ✓ Acknowledged

There is no guarantee for the user in withdrawal action that a sufficient amount of rAsko tokens that they should claim will be on the contract.

rAsko response

Not an issue. We have a require statement for this, this is a feature and enough rAsko will be allocated to farmers via our token economics.

C30I7a Deflationary tokens are not supported ● Medium ✓ Acknowledged

In the `deposit` function passed amount is added to the user's deposited amount. In the case where the LP-Token is a deflationary one, the passed amount and amount of tokens, which were actually received, will be different.

Recommendations

Consider supporting such kinds of tokens.

rAsko response

Not an issue. Deflationary tokens will not be supported our platform, our platform is not a regular master chef we will not be supporting every single token.

C30I7d Returned values of transfer are ignored ● Medium ✓ Acknowledged

Functions deposit and withdraw ignore return values by `rAsko.transfer` at L242 and L274. Several tokens do not revert in case of failure and return false.

rAsko response

Not an issue, only rAsko will be rewarded and will revert if transfer doesn't go through. Also we have the require statement for checking if there is enough balance.

C30I6b Immutable variable ● Low ✓ Resolved

Variable rAsko can be marked as immutable for the purpose of saving gas.

Update

The issue was fixed.

C30l6c Unused variable ● Low ✓ Resolved

Variable `rewardPerBlock` is declared, but not used.

Recommendation

Remove redundant declarations.

Update

The issue was fixed.

C30l6d Implicit visibility modifier ● Low ✓ Resolved

Variables `rAsko`, `apyCalculator` and `valueOf3000` are declared without a visibility modifier. It can make the code confusing.

Recommendation

Set visibility modifier explicitly each time. Even if it's supposed to be used as a default one.

Update

The issue was fixed. Modifiers were added.

C30l6e Lack of logging ● Low ✓ Partially fixed

There are no events in `dev()` and `setMigrator()` functions.

Update

The issue is marked as "Partially fixed" since function `setMigrator()` has been removed, but while the `dev()` function was not changed.

C30I6f Run out of gas possibility ● Low ✓ Acknowledged

In `massUpdatePools()` and `updateStakingPool()` functions there is a for-loop that iterates through all pools. If the amount of pools is large, the transaction may run out of gas.

Recommendation

To avoid the issue it's recommended to restrict the pool's amount or make the functions be able to work with slices of the array of the pool.

rAsko response

Not an issue there will be a limit to the amount of pools we will have it will not run out of gas.

C30I79 rAsko token cannot be in the pools ● Low ✓ Acknowledged

rAsko token cannot be in the pools since it gets the amount of tokens by calling `balanceOf` function at L207 to determine the total amount of deposit. It makes it impossible to use the rAsko token as an LP-Token. It also can be confusing since anyone can transfer LP-Tokens to the farming contract which changes one's balance without depositing.

rAsko response

Not an issue. Migrate is removed and dev() is not meant to emit an event.

C31. APYCalculator

Overview

An Oracle of the system.

Issues

C31I72 An Admin can't be removed ● High ✓ Resolved

There is no mechanism for removing addresses from the admin role. Some accounts can be stolen and there are no chances to remove them.

Recommendation

Define functionality of the removing APYCalculator's Admin role.

Update

The issue was fixed by removing ability setting more the one admin, which is set once.

rAsko response

Admin is set once and it will be the multi sig wallet. Furthermore this was never much of an issue to begin with since the apy calculator contract is changeable so if an apy contract is compromised a new one can easily be deployed in the first place

C31I7b Outdated oracle's data ● Medium ✓ Acknowledged

`APYCalculator`'s method `valueOf3000` just returns a value, but the functions don't check whether the value is actual. If the data will be outdated, data-consumer won't ever know that.

Recommendation

We recommend storing a timestamp when the value has been pushed to the contract storage and check if it was outdated in `valueOf3000`.

rAsko response

Not an issue if a price is the same after x time then change `ValueOf3000` will not be called to save gas therefore it will keep reverting even though the price is updated.

C32. rAskoSwapRouter

Overview

Implementation of the UniSwap Router expanding the one with fees on swap.

Issues

C32I75 Sending ETH by swap functions ● High ✓ Acknowledged

In functions `swapExactTokensForTokens()` L299 and `swapTokensForExactTokens()` L358 there are auto-send of ETH to the `DAOAddress`, `OperatingAddress` and `BuybackAddress` addresses appropriate commissions. It is a bad practice because on these addresses there could be a contract that doesn't implement `receive()` or `fallback()` function or it fails in certain situations. In this case, users couldn't make any swap through this router. In a worse case, these addresses could call the `assert()` function on receiving ETH and this could cause large consumption of gas from the user.

Recommendation

It is better to write in mapping how much ETH `DAOAddress`, `OperatingAddress` and `BuybackAddress` can withdraw and they will call the withdraw function by themselves.

rAsko response

This is not an issue these addresses are strictly wallet addresses not smart contract addresses

C32176 Locking ETH on the contract ● High ✓ Acknowledged

If a user sends more ETH in swap functions than required, this ETH will be lost in the contract. It also seems hard to calculate how many ETH should be exactly sent to perform swaps.

Recommendation

If fees must be payed in ETH, consider using WETH to perform transferFrom the call.

rAsko response

Not an issue since our front end will handle the calculations. This is standard anyways since in many swaps they give you an estimate of how many tokens you will receive via the view functions available in the library. We cannot use `WETH` as that is poor UX it requires users to carry around `WETH` which not many do

C32174 Transferring ETH by swap functions ● Medium ✓ Resolved

In functions `swapExactTokensForTokens()` L299 and `swapTokensForExactTokens()` L358 `transfer()` the function is used to send ETH. It is not recommended to use it for sending ETH.

Recommendation

It's better to use the `safeTransferETH()` function from the TransferHelper library.

Update

The issue was fixed by using the TransferHelper library.

C32I77 Unrestricted fees amount ● Medium ✓ Acknowledged

The owner of the contracts can set any amount of commission on swap (even 99.9%). Which is a serious risk for users interacting with the contract. This is related to the `DAOFee`, `OperatingFee` and `BuybackFee` too.

Recommendation

Restrict fees amount.

rAsko response

Not an issue. Firstly since the fees are taken in ETH the user clearly sees the `msg.value`. So they can choose to decline the transaction and its very visible. Furthermore the fees will be clearly stated and the governance will be setting the fees

C32I78 Inappropriate fees values ● Medium ✓ Acknowledged

In the router there is a possibility for the owner to change commissions on swap operations. But in pairs there is a fixed amount of minimum commission (it is 1%) and if the router calculates the wrong output amount (in case setting commissions lower than 1%) transaction performing swap through the router will fail. Also if the owner sets commission in the router bigger than 1%, it can be circumvented by the custom contract that uses only 1% as commission. Moreover commissions in ETH by this way also can be circumvented.

Update

The issue is not fixed. Function changeLPFee was removed, but it's still possible set an inappropriate fee value.

C32I7c Returned value of transferFrom is ignored ● Medium ✓ Acknowledged

The function `removeLiquidity` ignores the return value by `IPancakePair(pair).transferFrom` at L222. Several tokens do not revert in case of failure and return false.

rAsko response

Not an issue. Same with the farm on our front end only certain tokens will be available to swap and will revert when `transfer` or `transferFrom` fails.

C32I73 No events for admin functions ● Low ✓ Acknowledged

Functions, which are changing state do not have logging.

Recommendation

Emit events on setting up the state variables by calling admin functions.

Update

The issue is not fixed. In the new version of code variables LPFee, DAOFee and OperatingFee have been made public, but there are no events which can used as logging when and how the values were changed.

5. Conclusion

The audited contracts are strongly depend on Owner's behaviour.

The audited repository contains tests for the `rAskoSwapRouter` contract only. We strongly recommend adding tests for other contracts to ensure that the contracts work as intended.

Audit includes recommendations on the code improving and preventing potential attacks.

rAsko team response: Tests were originally commented for the farm test case file for faster testing on the swaps they have been uncommented

Update: the issues were addressed in the update. The issues were either fixed or team responses were added below them. Tests for rAskoFarm were uncommented, the main functionality is covered with tests.

Contracts are deployed to the Binance Smart Chain Mainnet:

rAsko: 0xd118f42edbc839f7e1e85d5269a25288792c141b

rAskoSwapFactory: 0xb38f27779cFa1507c813b417b24a5810F6A4E0A6

apyCalculator: 0x0f4298Af0d2B5b79cd35c448f7C2e5c75ACd5874

rAskoSwapRouter: 0xb77621d546c30526BCDdCd52A3B67C4a58F16d59

rAskoFarm: 0xeA65Df6D4a1D597b21CD96a8dAA816D3d0e428e2

Admin's address: 0xf8fD462Eeb4532a7b3cE8539315263AD6B2139ED

Appendix A. Issues’ severity classification

- **Critical.** Issues that may cause an unlimited loss of funds or entirely break the contract workflow. Malicious code (including malicious modification of libraries) is also treated as a critical severity issue. These issues must be fixed before deployments or fixed in already running projects as soon as possible.
- **High.** Issues that may lead to a limited loss of funds, break interaction with users, or other contracts under specific conditions. Also, issues in a smart contract, that allow a privileged account the ability to steal or block other users' funds.
- **Medium.** Issues that do not lead to a loss of funds directly, but break the contract logic. May lead to failures in contracts operation.
- **Low.** Issues that are of a non-optimal code character, for instance, gas optimization tips, unused variables, errors in messages.
- **Informational.** Issues that do not impact the contract operation. Usually, informational severity issues are related to code best practices, e.g. style guide.

Appendix B

- Business logic overview
- Functionality checks
- Following best practices
- Access control and authorization
- Reentrancy attacks
- Front-run attacks
- DoS with (unexpected) revert
- DoS with block gas limit
- Transaction-ordering dependence
- ERC/BEP and other standards violation
- Unchecked math
- Implicit visibility levels
- Excessive gas usage
- Timestamp dependence
- Forcibly sending ether to a contract
- Weak sources of randomness
- Shadowing state variables
- Usage of deprecated code

Appendix C. Listing of Slither output

```
rAskoSwapRouter.swapExactTokensForTokens(uint256,uint256,address[],address,uint256)
(contracts/swap/rAskoSwapRouter.sol#299-356) sends eth to arbitrary user
 Dangerous calls:
 - DAOAddress.transfer(DAORemainder) (contracts/swap/rAskoSwapRouter.sol#323)
 - OperatingAddress.transfer(OperatingRemainder) (contracts/swap/rAskoSwapRouter.sol#324)
 - BuybackAddress.transfer(BuybackRemainder) (contracts/swap/rAskoSwapRouter.sol#325)
 - DAOAddress.transfer(DAORemainder_scope_0) (contracts/swap/rAskoSwapRouter.sol#335)
 - OperatingAddress.transfer(OperatingRemainder_scope_1) (contracts/swap/
rAskoSwapRouter.sol#336)
 - BuybackAddress.transfer(BuybackRemainder_scope_2) (contracts/swap/
rAskoSwapRouter.sol#337)
rAskoSwapRouter.swapTokensForExactTokens(uint256,uint256,address[],address,uint256)
(contracts/swap/rAskoSwapRouter.sol#358-443) sends eth to arbitrary user
 Dangerous calls:
 - DAOAddress.transfer(DAORemainder) (contracts/swap/rAskoSwapRouter.sol#387)
 - OperatingAddress.transfer(OperatingRemainder) (contracts/swap/rAskoSwapRouter.sol#388)
 - BuybackAddress.transfer(BuybackRemainder) (contracts/swap/rAskoSwapRouter.sol#389)
 - DAOAddress.transfer(DAORemainder_scope_0) (contracts/swap/rAskoSwapRouter.sol#399)
 - OperatingAddress.transfer(OperatingRemainder_scope_1) (contracts/swap/
rAskoSwapRouter.sol#400)
 - BuybackAddress.transfer(BuybackRemainder_scope_2) (contracts/swap/
rAskoSwapRouter.sol#401)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#functions-that-
send-ether-to-arbitrary-destinations

rAskoSwapPair._update(uint256,uint256,uint112,uint112) (contracts/swap/
rAskoSwapPair.sol#73-86) uses a weak PRNG: "blockTimestamp = uint32(block.timestamp % 2 **
32) (contracts/swap/rAskoSwapPair.sol#75)"
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#weak-PRNG

IERC20 is re-used:
 - node_modules/@pancakeswap-libs/pancake-swap-core/contracts/interfaces/IERC20.sol#3-17
 - node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#8-77
 - node_modules/@theanthill/pancake-swap-periphery/contracts/interfaces/IERC20.sol#3-17
 - node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#8-77
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#name-reused

rAskoSwapRouter.removeLiquidity(address,address,uint256,uint256,uint256,address,uint256)
(contracts/swap/rAskoSwapRouter.sol#212-236) ignores return value by
```

```
IPancakePair(pair).transferFrom(msg.sender,pair,liquidity) (contracts/swap/
rAskoSwapRouter.sol#222)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-
transfer

rAskoFarm.deposit(uint256,uint256) (contracts/farm/rAskoFarm.sol#231-252) ignores return
value by rAsko.transfer(msg.sender,pending) (contracts/farm/rAskoFarm.sol#242)
rAskoFarm.withdraw(uint256,uint256) (contracts/farm/rAskoFarm.sol#255-282) ignores return
value by rAsko.transfer(msg.sender,pending) (contracts/farm/rAskoFarm.sol#274)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-
transfer

rAskoFarm.pendingReward(uint256,address) (contracts/farm/rAskoFarm.sol#179-190) performs a
multiplication on the result of a division:
 -reward = multiplier.mul(lpSupply).mul(10 ** 18).div(valueOf3000).mul(100).div(17280)
(contracts/farm/rAskoFarm.sol#186)
rAskoFarm.pendingReward(uint256,address) (contracts/farm/rAskoFarm.sol#179-190) performs a
multiplication on the result of a division:
 -reward = multiplier.mul(lpSupply).mul(10 ** 18).div(valueOf3000).mul(100).div(17280)
(contracts/farm/rAskoFarm.sol#186)
 -accRewardPerShare = accRewardPerShare.add(reward.mul(1e12).div(lpSupply)) (contracts/
farm/rAskoFarm.sol#187)
rAskoFarm.updatePool(uint256) (contracts/farm/rAskoFarm.sol#202-228) performs a
multiplication on the result of a division:
 -reward = multiplier.mul(lpSupply).mul(10 ** 18).div(valueOf3000).mul(100).div(17280)
(contracts/farm/rAskoFarm.sol#217)
rAskoFarm.updatePool(uint256) (contracts/farm/rAskoFarm.sol#202-228) performs a
multiplication on the result of a division:
 -reward = multiplier.mul(lpSupply).mul(10 ** 18).div(valueOf3000).mul(100).div(17280)
(contracts/farm/rAskoFarm.sol#217)
 -pool.accRewardPerShare = pool.accRewardPerShare.add(reward.mul(1e12).div(lpSupply))
(contracts/farm/rAskoFarm.sol#226)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-
multiply

rAskoSwapPair._safeTransfer(address,address,uint256) (contracts/swap/
rAskoSwapPair.sol#44-47) uses a dangerous strict equality:
 - require(bool,string)(success && (data.length == 0 || abi.decode(data,(bool))),Pancake:
TRANSFER_FAILED) (contracts/swap/rAskoSwapPair.sol#46)
rAskoSwapPair.mint(address) (contracts/swap/rAskoSwapPair.sol#110-131) uses a dangerous
strict equality:
 - _totalSupply == 0 (contracts/swap/rAskoSwapPair.sol#119)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-
equalities
```

```
rAskoFarm.migrate(uint256) (contracts/farm/rAskoFarm.sol#162-171) uses a dangerous strict
equality:
 - require(bool,string)(bal == newLpToken.balanceOf(address(this)),migrate: bad)
(contracts/farm/rAskoFarm.sol#169)
rAskoFarm.updatePool(uint256) (contracts/farm/rAskoFarm.sol#202-228) uses a dangerous
strict equality:
 - lpSupply == 0 (contracts/farm/rAskoFarm.sol#208)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-
equalities

Reentrancy in rAskoSwapPair.burn(address) (contracts/swap/rAskoSwapPair.sol#134-156):
 External calls:
 - _safeTransfer(_token0,to,amount0) (contracts/swap/rAskoSwapPair.sol#148)
  - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (contracts/swap/
rAskoSwapPair.sol#45)
 - _safeTransfer(_token1,to,amount1) (contracts/swap/rAskoSwapPair.sol#149)
  - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (contracts/swap/
rAskoSwapPair.sol#45)
 State variables written after the call(s):
 - _update(balance0,balance1,_reserve0,_reserve1) (contracts/swap/rAskoSwapPair.sol#153)
  - blockTimestampLast = blockTimestamp (contracts/swap/rAskoSwapPair.sol#84)
 - kLast = uint256(reserve0).mul(reserve1) (contracts/swap/rAskoSwapPair.sol#154)
 - _update(balance0,balance1,_reserve0,_reserve1) (contracts/swap/rAskoSwapPair.sol#153)
  - reserve0 = uint112(balance0) (contracts/swap/rAskoSwapPair.sol#82)
 - _update(balance0,balance1,_reserve0,_reserve1) (contracts/swap/rAskoSwapPair.sol#153)
  - reserve1 = uint112(balance1) (contracts/swap/rAskoSwapPair.sol#83)
Reentrancy in rAskoSwapFactory.createPair(address,address) (contracts/swap/
rAskoSwapFactory.sol#29-44):
 External calls:
 - IPancakePair(pair).initialize(token0,token1) (contracts/swap/rAskoSwapFactory.sol#39)
 State variables written after the call(s):
 - getPair[token0][token1] = pair (contracts/swap/rAskoSwapFactory.sol#40)
 - getPair[token1][token0] = pair (contracts/swap/rAskoSwapFactory.sol#41)
Reentrancy in rAskoSwapPair.swap(uint256,uint256,address,bytes) (contracts/swap/
rAskoSwapPair.sol#159-187):
 External calls:
 - _safeTransfer(_token0,to,amount0Out) (contracts/swap/rAskoSwapPair.sol#170)
  - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (contracts/swap/
rAskoSwapPair.sol#45)
 - _safeTransfer(_token1,to,amount1Out) (contracts/swap/rAskoSwapPair.sol#171)
  - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (contracts/swap/
rAskoSwapPair.sol#45)
```

```
 - IPancakeCallee(to).pancakeCall(msg.sender,amount0Out,amount1Out,data) (contracts/swap/
rAskoSwapPair.sol#172)
 State variables written after the call(s):
 - _update(balance0,balance1,_reserve0,_reserve1) (contracts/swap/rAskoSwapPair.sol#185)
  - blockTimestampLast = blockTimestamp (contracts/swap/rAskoSwapPair.sol#84)
 - _update(balance0,balance1,_reserve0,_reserve1) (contracts/swap/rAskoSwapPair.sol#185)
  - reserve0 = uint112(balance0) (contracts/swap/rAskoSwapPair.sol#82)
 - _update(balance0,balance1,_reserve0,_reserve1) (contracts/swap/rAskoSwapPair.sol#185)
  - reserve1 = uint112(balance1) (contracts/swap/rAskoSwapPair.sol#83)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-1

Reentrancy in rAskoFarm.deposit(uint256,uint256) (contracts/farm/rAskoFarm.sol#231-252):
 External calls:
 - rAsko.transfer(msg.sender,pending) (contracts/farm/rAskoFarm.sol#242)
 - pool.lpToken.safeTransferFrom(address(msg.sender),address(this),_amount) (contracts/
farm/rAskoFarm.sol#246)
 State variables written after the call(s):
 - user.amount = user.amount.add(_amount) (contracts/farm/rAskoFarm.sol#247)
 - user.depositBlock = block.number (contracts/farm/rAskoFarm.sol#248)
 - user.rewardDebt = user.amount.mul(pool.accRewardPerShare).div(1e12) (contracts/farm/
rAskoFarm.sol#250)
Reentrancy in rAskoFarm.emergencyWithdraw(uint256) (contracts/farm/rAskoFarm.sol#285-292):
 External calls:
 - pool.lpToken.safeTransfer(address(msg.sender),user.amount) (contracts/farm/
rAskoFarm.sol#288)
 State variables written after the call(s):
 - user.amount = 0 (contracts/farm/rAskoFarm.sol#290)
 - user.rewardDebt = 0 (contracts/farm/rAskoFarm.sol#291)
Reentrancy in rAskoFarm.migrate(uint256) (contracts/farm/rAskoFarm.sol#162-171):
 External calls:
 - lpToken.safeApprove(address(migrator),bal) (contracts/farm/rAskoFarm.sol#167)
 - newLpToken = migrator.migrate(lpToken) (contracts/farm/rAskoFarm.sol#168)
 State variables written after the call(s):
 - pool.lpToken = newLpToken (contracts/farm/rAskoFarm.sol#170)
Reentrancy in rAskoFarm.withdraw(uint256,uint256) (contracts/farm/rAskoFarm.sol#255-282):
 External calls:
 - rAsko.transfer(msg.sender,pending) (contracts/farm/rAskoFarm.sol#274)
 State variables written after the call(s):
 - user.amount = user.amount.sub(_amount) (contracts/farm/rAskoFarm.sol#277)
Reentrancy in rAskoFarm.withdraw(uint256,uint256) (contracts/farm/rAskoFarm.sol#255-282):
 External calls:
 - rAsko.transfer(msg.sender,pending) (contracts/farm/rAskoFarm.sol#274)
 - pool.lpToken.safeTransfer(address(msg.sender),_amount) (contracts/farm/
rAskoFarm.sol#278)
 State variables written after the call(s):
 - user.rewardDebt = user.amount.mul(pool.accRewardPerShare).div(1e12) (contracts/farm/
rAskoFarm.sol#280)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-1

rAskoSwapLibrary.getAmountsOut(address,uint256,address[],uint256).i (contracts/swap/
rAskoSwapLibrary.sol#130) is a local variable never initialized
rAskoSwapRouter._swap(uint256[],address[],address).i (contracts/swap/
rAskoSwapRouter.sol#280) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-
local-variables

rAskoSwapRouter._addLiquidity(address,address,uint256,uint256,uint256,uint256) (contracts/
swap/rAskoSwapRouter.sol#132-177) ignores return value by
IPancakeFactory(factory).createPair(tokenA,tokenB) (contracts/swap/
rAskoSwapRouter.sol#142)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

rAskoSwapRouter.changeAdmin(address) (contracts/swap/rAskoSwapRouter.sol#70-72) should
emit an event for:
 - admin = newAdmin (contracts/swap/rAskoSwapRouter.sol#71)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-
access-control

rAskoSwapRouter.changeLPFee(uint256) (contracts/swap/rAskoSwapRouter.sol#74-77) should
emit an event for:
 - LPFee = newFee (contracts/swap/rAskoSwapRouter.sol#76)
rAskoSwapRouter.changeDAOFee(uint256) (contracts/swap/rAskoSwapRouter.sol#79-85) should
emit an event for:
 - DAOFee = newFee (contracts/swap/rAskoSwapRouter.sol#84)
rAskoSwapRouter.changeOperatingFee(uint256) (contracts/swap/rAskoSwapRouter.sol#87-93)
should emit an event for:
 - OperatingFee = newFee (contracts/swap/rAskoSwapRouter.sol#92)
rAskoSwapRouter.changeBuybackFee(uint256) (contracts/swap/rAskoSwapRouter.sol#95-101)
should emit an event for:
 - BuybackFee = newFee (contracts/swap/rAskoSwapRouter.sol#100)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-
arithmetic

rAskoFarm.updateMultiplier(uint256) (contracts/farm/rAskoFarm.sol#101-103) should emit an
```

```
event for:
 - BONUS_MULTIPLIER = multiplierNumber (contracts/farm/rAskoFarm.sol#102)
rAskoFarm.add(uint256,IBEP20,bool) (contracts/farm/rAskoFarm.sol#115-128) should emit an
event for:
 - totalAllocPoint = totalAllocPoint.add(_allocPoint) (contracts/farm/rAskoFarm.sol#120)
rAskoFarm.set(uint256,uint256,bool) (contracts/farm/rAskoFarm.sol#131-141) should emit an
event for:
 - totalAllocPoint = totalAllocPoint.sub(prevAllocPoint).add(_allocPoint) (contracts/farm/
rAskoFarm.sol#138)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-
arithmetic

rAskoSwapFactory.constructor(address)._feeToSetter (contracts/swap/
rAskoSwapFactory.sol#17) lacks a zero-check on :
  - feeToSetter = _feeToSetter (contracts/swap/rAskoSwapFactory.sol#18)
rAskoSwapFactory.setFeeTo(address)._feeTo (contracts/swap/rAskoSwapFactory.sol#46) lacks a
zero-check on :
  - feeTo = _feeTo (contracts/swap/rAskoSwapFactory.sol#48)
rAskoSwapFactory.setFeeToSetter(address)._feeToSetter (contracts/swap/
rAskoSwapFactory.sol#51) lacks a zero-check on :
  - feeToSetter = _feeToSetter (contracts/swap/rAskoSwapFactory.sol#53)
rAskoSwapPair.initialize(address,address)._token0 (contracts/swap/rAskoSwapPair.sol#66)
lacks a zero-check on :
  - token0 = _token0 (contracts/swap/rAskoSwapPair.sol#68)
rAskoSwapPair.initialize(address,address)._token1 (contracts/swap/rAskoSwapPair.sol#66)
lacks a zero-check on :
  - token1 = _token1 (contracts/swap/rAskoSwapPair.sol#69)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-
address-validation

rAskoSwapRouter.constructor(address,uint256,uint256,address,uint256,address,uint256,address
,address,address)._factory (contracts/swap/rAskoSwapRouter.sol#46) lacks a zero-check on :
  - factory = _factory (contracts/swap/rAskoSwapRouter.sol#57)
rAskoSwapRouter.constructor(address,uint256,uint256,address,uint256,address,uint256,address
,address,address)._DAOAddress (contracts/swap/rAskoSwapRouter.sol#49) lacks a zero-check
on :
  - DAOAddress = address(_DAOAddress) (contracts/swap/rAskoSwapRouter.sol#60)
rAskoSwapRouter.constructor(address,uint256,uint256,address,uint256,address,uint256,address
,address,address)._OperatingAddress (contracts/swap/rAskoSwapRouter.sol#51) lacks a zero-
check on :
  - OperatingAddress = address(_OperatingAddress) (contracts/swap/rAskoSwapRouter.sol#62)
rAskoSwapRouter.constructor(address,uint256,uint256,address,uint256,address,uint256,address
,address,address)._BuybackAddress (contracts/swap/rAskoSwapRouter.sol#53) lacks a zero-
```

```
check on :
  - BuybackAddress = address(_BuybackAddress) (contracts/swap/rAskoSwapRouter.sol#64)
rAskoSwapRouter.constructor(address,uint256,uint256,address,uint256,address,uint256,address
,address,address)._baseLR (contracts/swap/rAskoSwapRouter.sol#54) lacks a zero-check on :
  - baseLR = _baseLR (contracts/swap/rAskoSwapRouter.sol#66)
rAskoSwapRouter.constructor(address,uint256,uint256,address,uint256,address,uint256,address
,address,address)._baseHR (contracts/swap/rAskoSwapRouter.sol#55) lacks a zero-check on :
  - baseHR = _baseHR (contracts/swap/rAskoSwapRouter.sol#67)
rAskoSwapRouter.changeAdmin(address).newAdmin (contracts/swap/rAskoSwapRouter.sol#70)
lacks a zero-check on :
  - admin = newAdmin (contracts/swap/rAskoSwapRouter.sol#71)
rAskoSwapRouter.changeBaseLR(address).newBaseLR (contracts/swap/rAskoSwapRouter.sol#103)
lacks a zero-check on :
  - baseLR = newBaseLR (contracts/swap/rAskoSwapRouter.sol#104)
rAskoSwapRouter.changeBaseHR(address).newBaseHR (contracts/swap/rAskoSwapRouter.sol#107)
lacks a zero-check on :
  - baseHR = newBaseHR (contracts/swap/rAskoSwapRouter.sol#108)
rAskoSwapRouter.changeDaoAddress(address).newDaoAddress (contracts/swap/
rAskoSwapRouter.sol#111) lacks a zero-check on :
  - DAOAddress = address(newDaoAddress) (contracts/swap/rAskoSwapRouter.sol#112)
rAskoSwapRouter.changeOperating(address).newOperatingAddress (contracts/swap/
rAskoSwapRouter.sol#115) lacks a zero-check on :
  - OperatingAddress = address(newOperatingAddress) (contracts/swap/
rAskoSwapRouter.sol#116)
rAskoSwapRouter.changeBuybackAddress(address).newBuybackAddress (contracts/swap/
rAskoSwapRouter.sol#119) lacks a zero-check on :
  - BuybackAddress = address(newBuybackAddress) (contracts/swap/rAskoSwapRouter.sol#120)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-
address-validation

rAskoFarm.constructor(IBEP20,IAPYCalculator,address,uint256,uint256)._devaddr (contracts/
farm/rAskoFarm.sol#90) lacks a zero-check on :
  - devaddr = _devaddr (contracts/farm/rAskoFarm.sol#96)
rAskoFarm.dev(address)._devaddr (contracts/farm/rAskoFarm.sol#295) lacks a zero-check on :
  - devaddr = _devaddr (contracts/farm/rAskoFarm.sol#297)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-
address-validation

rAskoSwapRouter._swap(uint256[],address[],address) (contracts/swap/
rAskoSwapRouter.sol#275-297) has external calls inside a loop: IPancakePair(rAskoSwapLibrar
y.pairFor(factory,input,output)).swap(amount0Out,amount1Out,to,new bytes(0)) (contracts/
swap/rAskoSwapRouter.sol#290-295)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-
loop
```

```
Reentrancy in rAskoSwapPair.burn(address) (contracts/swap/rAskoSwapPair.sol#134-156):
 External calls:
 - _safeTransfer(_token0,to,amount0) (contracts/swap/rAskoSwapPair.sol#148)
  - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (contracts/swap/
rAskoSwapPair.sol#45)
 - _safeTransfer(_token1,to,amount1) (contracts/swap/rAskoSwapPair.sol#149)
  - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (contracts/swap/
rAskoSwapPair.sol#45)
 State variables written after the call(s):
 - _update(balance0,balance1,_reserve0,_reserve1) (contracts/swap/rAskoSwapPair.sol#153)
  - price0CumulativeLast += uint256(UQ112x112.encode(_reserve1).uqdiv(_reserve0)) *
timeElapsed (contracts/swap/rAskoSwapPair.sol#79)
 - _update(balance0,balance1,_reserve0,_reserve1) (contracts/swap/rAskoSwapPair.sol#153)
  - price1CumulativeLast += uint256(UQ112x112.encode(_reserve0).uqdiv(_reserve1)) *
timeElapsed (contracts/swap/rAskoSwapPair.sol#80)
Reentrancy in rAskoSwapFactory.createPair(address,address) (contracts/swap/
rAskoSwapFactory.sol#29-44):
 External calls:
 - IPancakePair(pair).initialize(token0,token1) (contracts/swap/rAskoSwapFactory.sol#39)
 State variables written after the call(s):
 - allPairs.push(pair) (contracts/swap/rAskoSwapFactory.sol#42)
Reentrancy in rAskoSwapPair.swap(uint256,uint256,address,bytes) (contracts/swap/
rAskoSwapPair.sol#159-187):
 External calls:
 - _safeTransfer(_token0,to,amount0Out) (contracts/swap/rAskoSwapPair.sol#170)
  - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (contracts/swap/
rAskoSwapPair.sol#45)
 - _safeTransfer(_token1,to,amount1Out) (contracts/swap/rAskoSwapPair.sol#171)
  - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (contracts/swap/
rAskoSwapPair.sol#45)
 - IPancakeCallee(to).pancakeCall(msg.sender,amount0Out,amount1Out,data) (contracts/swap/
rAskoSwapPair.sol#172)
 State variables written after the call(s):
 - _update(balance0,balance1,_reserve0,_reserve1) (contracts/swap/rAskoSwapPair.sol#185)
  - price0CumulativeLast += uint256(UQ112x112.encode(_reserve1).uqdiv(_reserve0)) *
timeElapsed (contracts/swap/rAskoSwapPair.sol#79)
 - _update(balance0,balance1,_reserve0,_reserve1) (contracts/swap/rAskoSwapPair.sol#185)
  - price1CumulativeLast += uint256(UQ112x112.encode(_reserve0).uqdiv(_reserve1)) *
timeElapsed (contracts/swap/rAskoSwapPair.sol#80)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-2
```

```
Reentrancy in rAskoSwapPair.burn(address) (contracts/swap/rAskoSwapPair.sol#134-156):
 External calls:
 - _safeTransfer(_token0,to,amount0) (contracts/swap/rAskoSwapPair.sol#148)
  - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (contracts/swap/
rAskoSwapPair.sol#45)
 - _safeTransfer(_token1,to,amount1) (contracts/swap/rAskoSwapPair.sol#149)
  - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (contracts/swap/
rAskoSwapPair.sol#45)
 Event emitted after the call(s):
 - Burn(msg.sender,amount0,amount1,to) (contracts/swap/rAskoSwapPair.sol#155)
 - Sync(reserve0,reserve1) (contracts/swap/rAskoSwapPair.sol#85)
  - _update(balance0,balance1,_reserve0,_reserve1) (contracts/swap/rAskoSwapPair.sol#153)
Reentrancy in rAskoSwapFactory.createPair(address,address) (contracts/swap/
rAskoSwapFactory.sol#29-44):
 External calls:
 - IPancakePair(pair).initialize(token0,token1) (contracts/swap/rAskoSwapFactory.sol#39)
 Event emitted after the call(s):
 - PairCreated(token0,token1,pair,allPairs.length) (contracts/swap/
rAskoSwapFactory.sol#43)
Reentrancy in rAskoSwapPair.swap(uint256,uint256,address,bytes) (contracts/swap/
rAskoSwapPair.sol#159-187):
 External calls:
 - _safeTransfer(_token0,to,amount0Out) (contracts/swap/rAskoSwapPair.sol#170)
  - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (contracts/swap/
rAskoSwapPair.sol#45)
 - _safeTransfer(_token1,to,amount1Out) (contracts/swap/rAskoSwapPair.sol#171)
  - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (contracts/swap/
rAskoSwapPair.sol#45)
 - IPancakeCallee(to).pancakeCall(msg.sender,amount0Out,amount1Out,data) (contracts/swap/
rAskoSwapPair.sol#172)
 Event emitted after the call(s):
 - Swap(msg.sender,amount0In,amount1In,amount0Out,amount1Out,to) (contracts/swap/
rAskoSwapPair.sol#186)
 - Sync(reserve0,reserve1) (contracts/swap/rAskoSwapPair.sol#85)
  - _update(balance0,balance1,_reserve0,_reserve1) (contracts/swap/rAskoSwapPair.sol#185)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-3

Reentrancy in rAskoFarm.deposit(uint256,uint256) (contracts/farm/rAskoFarm.sol#231-252):
 External calls:
 - rAsko.transfer(msg.sender,pending) (contracts/farm/rAskoFarm.sol#242)
 - pool.lpToken.safeTransferFrom(address(msg.sender),address(this),_amount) (contracts/
farm/rAskoFarm.sol#246)
```

```
 Event emitted after the call(s):
 - Deposit(msg.sender,_pid,_amount) (contracts/farm/rAskoFarm.sol#251)
Reentrancy in rAskoFarm.emergencyWithdraw(uint256) (contracts/farm/rAskoFarm.sol#285-292):
 External calls:
 - pool.lpToken.safeTransfer(address(msg.sender),user.amount) (contracts/farm/
rAskoFarm.sol#288)
 Event emitted after the call(s):
 - EmergencyWithdraw(msg.sender,_pid,user.amount) (contracts/farm/rAskoFarm.sol#289)
Reentrancy in rAskoFarm.withdraw(uint256,uint256) (contracts/farm/rAskoFarm.sol#255-282):
 External calls:
 - rAsko.transfer(msg.sender,pending) (contracts/farm/rAskoFarm.sol#274)
 - pool.lpToken.safeTransfer(address(msg.sender),_amount) (contracts/farm/
rAskoFarm.sol#278)
 Event emitted after the call(s):
 - Withdraw(msg.sender,_pid,_amount) (contracts/farm/rAskoFarm.sol#281)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-3

Dai.permit(address,address,uint256,uint256,bool,uint8,bytes32,bytes32) (contracts/tokens/
DAI.sol#171-193) uses timestamp for comparisons
 Dangerous comparisons:
 - require(bool,string)(expiry == 0 || now <= expiry,Dai/permit-expired) (contracts/tokens/
DAI.sol#188)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

PancakeERC20.permit(address,address,uint256,uint256,uint8,bytes32,bytes32) (node_modules/
@pancakeswap-libs/pancake-swap-core/contracts/PancakeERC20.sol#81-93) uses timestamp for
comparisons
 Dangerous comparisons:
 - require(bool,string)(deadline >= block.timestamp,Pancake: EXPIRED) (node_modules/
@pancakeswap-libs/pancake-swap-core/contracts/PancakeERC20.sol#82)
rAskoSwapPair._update(uint256,uint256,uint112,uint112) (contracts/swap/
rAskoSwapPair.sol#73-86) uses timestamp for comparisons
 Dangerous comparisons:
 - timeElapsed > 0 && _reserve0 != 0 && _reserve1 != 0 (contracts/swap/
rAskoSwapPair.sol#77)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

PancakeERC20.constructor() (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/
PancakeERC20.sol#24-38) uses assembly
 - INLINE ASM (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/
PancakeERC20.sol#26-28)
rAskoSwapFactory.createPair(address,address) (contracts/swap/rAskoSwapFactory.sol#29-44)
```

```
uses assembly
 - INLINE ASM (contracts/swap/rAskoSwapFactory.sol#36-38)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

console._sendLogPayload(bytes) (node_modules/hardhat/console.sol#7-14) uses assembly
 - INLINE ASM (node_modules/hardhat/console.sol#10-13)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Address.isContract(address) (node_modules/@pancakeswap/pancake-swap-lib/contracts/utils/
Address.sol#26-37) uses assembly
 - INLINE ASM (node_modules/@pancakeswap/pancake-swap-lib/contracts/utils/
Address.sol#33-35)
Address._functionCallWithValue(address,bytes,uint256,string) (node_modules/@pancakeswap/
pancake-swap-lib/contracts/utils/Address.sol#134-160) uses assembly
 - INLINE ASM (node_modules/@pancakeswap/pancake-swap-lib/contracts/utils/
Address.sol#152-155)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Different versions of Solidity is used:
 - Version used: ['=0.5.16', '>=0.5.0']
 - =0.5.16 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/PancakeERC20.sol#1)
 - >=0.5.0 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/interfaces/
IERC20.sol#1)
 - >=0.5.0 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/interfaces/
IPancakeCallee.sol#1)
 - >=0.5.0 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/interfaces/
IPancakeERC20.sol#1)
 - >=0.5.0 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/interfaces/
IPancakeFactory.sol#1)
 - >=0.5.0 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/interfaces/
IPancakePair.sol#1)
 - =0.5.16 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/libraries/
Math.sol#1)
 - =0.5.16 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/libraries/
SafeMath.sol#1)
 - =0.5.16 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/libraries/
UQ112x112.sol#1)
 - =0.5.16 (contracts/swap/rAskoSwapFactory.sol#1)
 - =0.5.16 (contracts/swap/rAskoSwapPair.sol#1)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-
directives-are-used

Different versions of Solidity is used:
 - Version used: ['=0.6.6', '>=0.4.22<0.9.0', '>=0.5.0', '>=0.6.0', '>=0.6.2']
 - >=0.5.0 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/interfaces/
IPancakeFactory.sol#1)
 - >=0.5.0 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/interfaces/
IPancakePair.sol#1)
 - >=0.5.0 (node_modules/@theanthill/pancake-swap-periphery/contracts/interfaces/
IERC20.sol#1)
 - >=0.6.0 (node_modules/@uniswap/lib/contracts/libraries/TransferHelper.sol#3)
 - >=0.6.2 (contracts/swap/interfaces/IrAskoSwapRouter.sol#1)
 - >=0.5.0 (contracts/swap/rAskoSwapLibrary.sol#1)
 - =0.6.6 (contracts/swap/rAskoSwapRouter.sol#1)
 - =0.6.6 (contracts/utils/SafeMath.sol#1)
 - >=0.4.22<0.9.0 (node_modules/hardhat/console.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-
directives-are-used

Different versions of Solidity is used:
 - Version used: ['0.6.12', '>=0.4.0', '>=0.4.22<0.9.0', '^0.6.0', '^0.6.12', '^0.6.2']
 - >=0.4.0 (node_modules/@pancakeswap/pancake-swap-lib/contracts/GSN/Context.sol#3)
 - >=0.4.0 (node_modules/@pancakeswap/pancake-swap-lib/contracts/access/Ownable.sol#3)
 - >=0.4.0 (node_modules/@pancakeswap/pancake-swap-lib/contracts/math/SafeMath.sol#3)
 - >=0.4.0 (node_modules/@pancakeswap/pancake-swap-lib/contracts/token/BEP20/IBEP20.sol#3)
 - ^0.6.0 (node_modules/@pancakeswap/pancake-swap-lib/contracts/token/BEP20/
SafeBEP20.sol#3)
 - ^0.6.2 (node_modules/@pancakeswap/pancake-swap-lib/contracts/utils/Address.sol#3)
 - ^0.6.12 (contracts/farm/interfaces/IAPYCalculator.sol#1)
 - 0.6.12 (contracts/farm/rAskoFarm.sol#1)
 - >=0.4.22<0.9.0 (node_modules/hardhat/console.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-
directives-are-used

Different versions of Solidity is used:
 - Version used: ['>=0.4.0', '>=0.4.22<0.9.0', '>=0.5.0', '>=0.6.0', '>=0.6.2', '^0.8.0']
 - ^0.8.0 (node_modules/@openzeppelin/contracts/access/AccessControl.sol#3)
 - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#3)
 - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#3)
 - ^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/
IERC20Metadata.sol#3)
 - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#3)
 - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/Strings.sol#3)
 - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/ERC165.sol#3)
 - ^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/IERC165.sol#3)
 - >=0.5.0 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/interfaces/
IERC20.sol#1)
```

```
 - >=0.5.0 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/interfaces/
IPancakeCallee.sol#1)
 - >=0.5.0 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/interfaces/
IPancakeERC20.sol#1)
 - >=0.5.0 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/interfaces/
IPancakeFactory.sol#1)
 - >=0.5.0 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/interfaces/
IPancakePair.sol#1)
 - >=0.4.0 (node_modules/@pancakeswap/pancake-swap-lib/contracts/math/SafeMath.sol#3)
 - >=0.4.0 (node_modules/@pancakeswap/pancake-swap-lib/contracts/token/BEP20/IBEP20.sol#3)
 - >=0.5.0 (node_modules/@theanthill/pancake-swap-periphery/contracts/interfaces/
IERC20.sol#1)
 - >=0.6.0 (node_modules/@uniswap/lib/contracts/libraries/TransferHelper.sol#3)
 - ^0.8.0 (contracts/farm/APYCalculator.sol#1)
 - >=0.6.2 (contracts/swap/interfaces/IrAskoSwapRouter.sol#1)
 - ^0.8.0 (contracts/tokens/MOKToken.sol#2)
 - ^0.8.0 (contracts/tokens/rASKO.sol#1)
 - >=0.4.22<0.9.0 (node_modules/hardhat/console.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-
directives-are-used

Pragma version^0.4.18 (contracts/tokens/WETH.sol#1) allows old versions
solc-0.4.18 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity

Pragma version=0.5.12 (contracts/tokens/DAI.sol#6) allows old versions
solc-0.5.12 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity

Pragma version>=0.5.0 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/
interfaces/IERC20.sol#1) allows old versions
Pragma version>=0.5.0 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/
interfaces/IPancakeCallee.sol#1) allows old versions
Pragma version>=0.5.0 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/
interfaces/IPancakeERC20.sol#1) allows old versions
Pragma version>=0.5.0 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/
interfaces/IPancakeFactory.sol#1) allows old versions
Pragma version>=0.5.0 (node_modules/@pancakeswap-libs/pancake-swap-core/contracts/
interfaces/IPancakePair.sol#1) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity
```

```
Pragma version>=0.5.0 (node_modules/@theanthill/pancake-swap-periphery/contracts/
interfaces/IERC20.sol#1) allows old versions
Pragma version>=0.6.0 (node_modules/@uniswap/lib/contracts/libraries/TransferHelper.sol#3)
allows old versions
Pragma version>=0.6.2 (contracts/swap/interfaces/IrAskoSwapRouter.sol#1) allows old
versions
Pragma version>=0.5.0 (contracts/swap/rAskoSwapLibrary.sol#1) allows old versions
Pragma version=0.6.6 (contracts/swap/rAskoSwapRouter.sol#1) allows old versions
Pragma version=0.6.6 (contracts/utils/SafeMath.sol#1) allows old versions
Pragma version>=0.4.22<0.9.0 (node_modules/hardhat/console.sol#2) is too complex
solc-0.6.6 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity

Pragma version>=0.4.0 (node_modules/@pancakeswap/pancake-swap-lib/contracts/GSN/
Context.sol#3) allows old versions
Pragma version>=0.4.0 (node_modules/@pancakeswap/pancake-swap-lib/contracts/access/
Ownable.sol#3) allows old versions
Pragma version>=0.4.0 (node_modules/@pancakeswap/pancake-swap-lib/contracts/math/
SafeMath.sol#3) allows old versions
Pragma version>=0.4.0 (node_modules/@pancakeswap/pancake-swap-lib/contracts/token/BEP20/
IBEP20.sol#3) allows old versions
Pragma version^0.6.0 (node_modules/@pancakeswap/pancake-swap-lib/contracts/token/BEP20/
SafeBEP20.sol#3) allows old versions
Pragma version^0.6.2 (node_modules/@pancakeswap/pancake-swap-lib/contracts/utils/
Address.sol#3) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity

Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/access/AccessControl.sol#3)
necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#3)
necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/IERC20.sol#3)
necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/token/ERC20/extensions/
IERC20Metadata.sol#3) necessitates a version too recent to be trusted. Consider deploying
with 0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Context.sol#3)
necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/Strings.sol#3)
necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
```

```
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/
ERC165.sol#3) necessitates a version too recent to be trusted. Consider deploying with
0.6.12/0.7.6
Pragma version^0.8.0 (node_modules/@openzeppelin/contracts/utils/introspection/
IERC165.sol#3) necessitates a version too recent to be trusted. Consider deploying with
0.6.12/0.7.6
Pragma version^0.8.0 (contracts/farm/APYCalculator.sol#1) necessitates a version too
recent to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (contracts/tokens/MOKToken.sol#2) necessitates a version too recent
to be trusted. Consider deploying with 0.6.12/0.7.6
Pragma version^0.8.0 (contracts/tokens/rASKO.sol#1) necessitates a version too recent to
be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.0 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity

Low level call in rAskoSwapPair._safeTransfer(address,address,uint256) (contracts/swap/
rAskoSwapPair.sol#44-47):
 - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (contracts/swap/
rAskoSwapPair.sol#45)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Low level call in TransferHelper.safeApprove(address,address,uint256) (node_modules/
@uniswap/lib/contracts/libraries/TransferHelper.sol#7-18):
 - (success,data) = token.call(abi.encodeWithSelector(0x095ea7b3,to,value)) (node_modules/
@uniswap/lib/contracts/libraries/TransferHelper.sol#13)
Low level call in TransferHelper.safeTransfer(address,address,uint256) (node_modules/
@uniswap/lib/contracts/libraries/TransferHelper.sol#20-31):
 - (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,value)) (node_modules/
@uniswap/lib/contracts/libraries/TransferHelper.sol#26)
Low level call in TransferHelper.safeTransferFrom(address,address,address,uint256)
(node_modules/@uniswap/lib/contracts/libraries/TransferHelper.sol#33-45):
 - (success,data) = token.call(abi.encodeWithSelector(0x23b872dd,from,to,value))
(node_modules/@uniswap/lib/contracts/libraries/TransferHelper.sol#40)
Low level call in TransferHelper.safeTransferETH(address,uint256) (node_modules/@uniswap/
lib/contracts/libraries/TransferHelper.sol#47-50):
 - (success) = to.call{value: value}(new bytes(0)) (node_modules/@uniswap/lib/contracts/
libraries/TransferHelper.sol#48)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Low level call in Address.sendValue(address,uint256) (node_modules/@pancakeswap/pancake-
swap-lib/contracts/utils/Address.sol#55-61):
 - (success) = recipient.call{value: amount}() (node_modules/@pancakeswap/pancake-swap-lib/
contracts/utils/Address.sol#59)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string)
(node_modules/@pancakeswap/pancake-swap-lib/contracts/utils/Address.sol#134-160):
 - (success,returndata) = target.call{value: weiValue}(data) (node_modules/@pancakeswap/
pancake-swap-lib/contracts/utils/Address.sol#143)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Constant Dai.version (contracts/tokens/DAI.sol#86) is not in UPPER_CASE_WITH_UNDERSCORES
Variable Dai.DOMAIN_SEPARATOR (contracts/tokens/DAI.sol#106) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-
solidity-naming-conventions

Variable PancakeERC20.DOMAIN_SEPARATOR (node_modules/@pancakeswap-libs/pancake-swap-core/
contracts/PancakeERC20.sol#16) is not in mixedCase
Function IPancakeERC20.DOMAIN_SEPARATOR() (node_modules/@pancakeswap-libs/pancake-swap-
core/contracts/interfaces/IPancakeERC20.sol#18) is not in mixedCase
Function IPancakeERC20.PERMIT_TYPEHASH() (node_modules/@pancakeswap-libs/pancake-swap-core/
contracts/interfaces/IPancakeERC20.sol#19) is not in mixedCase
Function IPancakePair.DOMAIN_SEPARATOR() (node_modules/@pancakeswap-libs/pancake-swap-core/
contracts/interfaces/IPancakePair.sol#18) is not in mixedCase
Function IPancakePair.PERMIT_TYPEHASH() (node_modules/@pancakeswap-libs/pancake-swap-core/
contracts/interfaces/IPancakePair.sol#19) is not in mixedCase
Function IPancakePair.MINIMUM_LIQUIDITY() (node_modules/@pancakeswap-libs/pancake-swap-
core/contracts/interfaces/IPancakePair.sol#36) is not in mixedCase
Contract rAskoSwapFactory (contracts/swap/rAskoSwapFactory.sol#6-56) is not in CapWords
Parameter rAskoSwapFactory.setFeeTo(address)._feeTo (contracts/swap/
rAskoSwapFactory.sol#46) is not in mixedCase
Parameter rAskoSwapFactory.setFeeToSetter(address)._feeToSetter (contracts/swap/
rAskoSwapFactory.sol#51) is not in mixedCase
Contract rAskoSwapPair (contracts/swap/rAskoSwapPair.sol#11-202) is not in CapWords
Parameter rAskoSwapPair.initialize(address,address)._token0 (contracts/swap/
rAskoSwapPair.sol#66) is not in mixedCase
Parameter rAskoSwapPair.initialize(address,address)._token1 (contracts/swap/
rAskoSwapPair.sol#66) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-
solidity-naming-conventions

Contract rAskoSwapLibrary (contracts/swap/rAskoSwapLibrary.sol#8-169) is not in CapWords
Contract rAskoSwapRouter (contracts/swap/rAskoSwapRouter.sol#13-496) is not in CapWords
Variable rAskoSwapRouter.LPFee (contracts/swap/rAskoSwapRouter.sol#17) is not in mixedCase
Variable rAskoSwapRouter.DAOFee (contracts/swap/rAskoSwapRouter.sol#19) is not in
mixedCase
Variable rAskoSwapRouter.DAOAddress (contracts/swap/rAskoSwapRouter.sol#20) is not in
mixedCase
```

```
Variable rAskoSwapRouter.OperatingFee (contracts/swap/rAskoSwapRouter.sol#22) is not in
mixedCase
Variable rAskoSwapRouter.OperatingAddress (contracts/swap/rAskoSwapRouter.sol#23) is not
in mixedCase
Variable rAskoSwapRouter.BuybackFee (contracts/swap/rAskoSwapRouter.sol#25) is not in
mixedCase
Variable rAskoSwapRouter.BuybackAddress (contracts/swap/rAskoSwapRouter.sol#26) is not in
mixedCase
Contract console (node_modules/hardhat/console.sol#4-1532) is not in CapWords
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-
solidity-naming-conventions

Contract rAskoFarm (contracts/farm/rAskoFarm.sol#27-300) is not in CapWords
Parameter rAskoFarm.changeApyCalculator(address)._apyCalculator (contracts/farm/
rAskoFarm.sol#105) is not in mixedCase
Parameter rAskoFarm.add(uint256,IBEP20,bool)._allocPoint (contracts/farm/
rAskoFarm.sol#115) is not in mixedCase
Parameter rAskoFarm.add(uint256,IBEP20,bool)._lpToken (contracts/farm/rAskoFarm.sol#115)
is not in mixedCase
Parameter rAskoFarm.add(uint256,IBEP20,bool)._withUpdate (contracts/farm/
rAskoFarm.sol#115) is not in mixedCase
Parameter rAskoFarm.set(uint256,uint256,bool)._pid (contracts/farm/rAskoFarm.sol#131) is
not in mixedCase
Parameter rAskoFarm.set(uint256,uint256,bool)._allocPoint (contracts/farm/
rAskoFarm.sol#131) is not in mixedCase
Parameter rAskoFarm.set(uint256,uint256,bool)._withUpdate (contracts/farm/
rAskoFarm.sol#131) is not in mixedCase
Parameter rAskoFarm.setMigrator(IMigratorChef)._migrator (contracts/farm/
rAskoFarm.sol#157) is not in mixedCase
Parameter rAskoFarm.migrate(uint256)._pid (contracts/farm/rAskoFarm.sol#162) is not in
mixedCase
Parameter rAskoFarm.getMultiplier(uint256,uint256)._from (contracts/farm/
rAskoFarm.sol#174) is not in mixedCase
Parameter rAskoFarm.getMultiplier(uint256,uint256)._to (contracts/farm/rAskoFarm.sol#174)
is not in mixedCase
Parameter rAskoFarm.pendingReward(uint256,address)._pid (contracts/farm/rAskoFarm.sol#179)
is not in mixedCase
Parameter rAskoFarm.pendingReward(uint256,address)._user (contracts/farm/
rAskoFarm.sol#179) is not in mixedCase
Parameter rAskoFarm.updatePool(uint256)._pid (contracts/farm/rAskoFarm.sol#202) is not in
mixedCase
Parameter rAskoFarm.deposit(uint256,uint256)._pid (contracts/farm/rAskoFarm.sol#231) is
not in mixedCase
```

```
Parameter rAskoFarm.deposit(uint256,uint256)._amount (contracts/farm/rAskoFarm.sol#231) is
not in mixedCase
Parameter rAskoFarm.withdraw(uint256,uint256)._pid (contracts/farm/rAskoFarm.sol#255) is
not in mixedCase
Parameter rAskoFarm.withdraw(uint256,uint256)._amount (contracts/farm/rAskoFarm.sol#255)
is not in mixedCase
Parameter rAskoFarm.emergencyWithdraw(uint256)._pid (contracts/farm/rAskoFarm.sol#285) is
not in mixedCase
Parameter rAskoFarm.dev(address)._devaddr (contracts/farm/rAskoFarm.sol#295) is not in
mixedCase
Variable rAskoFarm.BONUS_MULTIPLIER (contracts/farm/rAskoFarm.sol#70) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-
solidity-naming-conventions

Constant Strings.alphabet (node_modules/@openzeppelin/contracts/utils/Strings.sol#9) is
not in UPPER_CASE_WITH_UNDERSCORES
Parameter APYCalculator.changeValues(address[],uint256[])._amounts (contracts/farm/
APYCalculator.sol#31) is not in mixedCase
Contract rASKO (contracts/tokens/rASKO.sol#5-10) is not in CapWords
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-
solidity-naming-conventions

Redundant expression "this (node_modules/@pancakeswap/pancake-swap-lib/contracts/GSN/
Context.sol#25)" inContext (node_modules/@pancakeswap/pancake-swap-lib/contracts/GSN/
Context.sol#15-28)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-
statements

Redundant expression "this (node_modules/@openzeppelin/contracts/utils/Context.sol#21)"
inContext (node_modules/@openzeppelin/contracts/utils/Context.sol#15-24)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-
statements

Reentrancy in WETH9.withdraw(uint256) (contracts/tokens/WETH.sol#23-28):
 External calls:
 - msg.sender.transfer(wad) (contracts/tokens/WETH.sol#26)
 Event emitted after the call(s):
 - Withdrawal(msg.sender,wad) (contracts/tokens/WETH.sol#27)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-4

Variable rAskoSwapPair.swap(uint256,uint256,address,bytes).balance0Adjusted (contracts/
swap/rAskoSwapPair.sol#180) is too similar to
```

```
rAskoSwapPair.swap(uint256,uint256,address,bytes).balance1Adjusted (contracts/swap/
rAskoSwapPair.sol#181)
Variable rAskoSwapPair.price0CumulativeLast (contracts/swap/rAskoSwapPair.sol#26) is too
similar to rAskoSwapPair.price1CumulativeLast (contracts/swap/rAskoSwapPair.sol#27)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-
are-too-similar

Variable IrAskoSwapRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,addr
ess,uint256).amountADesired (contracts/swap/interfaces/IrAskoSwapRouter.sol#9) is too
similar to IrAskoSwapRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,ad
dress,uint256).amountBDesired (contracts/swap/interfaces/IrAskoSwapRouter.sol#10)
Variable rAskoSwapRouter._addLiquidity(address,address,uint256,uint256,uint256,uint256).amo
untADesired (contracts/swap/rAskoSwapRouter.sol#135) is too similar to rAskoSwapRouter._add
Liquidity(address,address,uint256,uint256,uint256,uint256).amountBDesired (contracts/swap/
rAskoSwapRouter.sol#136)
Variable rAskoSwapRouter._addLiquidity(address,address,uint256,uint256,uint256,uint256).amo
untADesired (contracts/swap/rAskoSwapRouter.sol#135) is too similar to rAskoSwapRouter.addL
iquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired
(contracts/swap/rAskoSwapRouter.sol#183)
Variable rAskoSwapRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,addre
ss,uint256).amountADesired (contracts/swap/rAskoSwapRouter.sol#182) is too similar to rAsko
SwapRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).am
ountBDesired (contracts/swap/rAskoSwapRouter.sol#183)
Variable rAskoSwapRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,addre
ss,uint256).amountADesired (contracts/swap/rAskoSwapRouter.sol#182) is too similar to rAsko
SwapRouter._addLiquidity(address,address,uint256,uint256,uint256,uint256).amountBDesired
(contracts/swap/rAskoSwapRouter.sol#136)
Variable rAskoSwapRouter._addLiquidity(address,address,uint256,uint256,uint256,uint256).amo
untAOptimal (contracts/swap/rAskoSwapRouter.sol#164-168) is too similar to rAskoSwapRouter.
_addLiquidity(address,address,uint256,uint256,uint256,uint256).amountBOptimal (contracts/
swap/rAskoSwapRouter.sol#152-156)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-
are-too-similar

rAskoSwapFactory.createPair(address,address) (contracts/swap/rAskoSwapFactory.sol#29-44)
uses literals with too many digits:
 - bytecode = type(address)(rAskoSwapPair).creationCode (contracts/swap/
rAskoSwapFactory.sol#34)
rAskoSwapFactory.slitherConstructorConstantVariables() (contracts/swap/
rAskoSwapFactory.sol#6-56) uses literals with too many digits:
 - INIT_CODE_PAIR_HASH = keccak256(bytes)(abi.encodePacked(type(address)
(rAskoSwapPair).creationCode)) (contracts/swap/rAskoSwapFactory.sol#7)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
```

```
console.slitherConstructorConstantVariables() (node_modules/hardhat/console.sol#4-1532)
uses literals with too many digits:
 - CONSOLE_ADDRESS = address(0x000000000000000000636F6e736F6c652e6c6f67) (node_modules/
hardhat/console.sol#5)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

WETH9.decimals (contracts/tokens/WETH.sol#6) should be constant
WETH9.name (contracts/tokens/WETH.sol#4) should be constant
WETH9.symbol (contracts/tokens/WETH.sol#5) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-
that-could-be-declared-constant

fallback() should be declared external:
 - WETH9.fallback() (contracts/tokens/WETH.sol#16-18)
withdraw(uint256) should be declared external:
 - WETH9.withdraw(uint256) (contracts/tokens/WETH.sol#23-28)
totalSupply() should be declared external:
 - WETH9.totalSupply() (contracts/tokens/WETH.sol#30-32)
approve(address,uint256) should be declared external:
 - WETH9.approve(address,uint256) (contracts/tokens/WETH.sol#34-38)
transfer(address,uint256) should be declared external:
 - WETH9.transfer(address,uint256) (contracts/tokens/WETH.sol#40-42)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-
that-could-be-declared-external

getHash() should be declared external:
 - rAskoSwapFactory.getHash() (contracts/swap/rAskoSwapFactory.sol#21-23)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-
that-could-be-declared-external

changeAdmin(address) should be declared external:
 - rAskoSwapRouter.changeAdmin(address) (contracts/swap/rAskoSwapRouter.sol#70-72)
changeLPFee(uint256) should be declared external:
 - rAskoSwapRouter.changeLPFee(uint256) (contracts/swap/rAskoSwapRouter.sol#74-77)
changeDAOFee(uint256) should be declared external:
 - rAskoSwapRouter.changeDAOFee(uint256) (contracts/swap/rAskoSwapRouter.sol#79-85)
changeOperatingFee(uint256) should be declared external:
 - rAskoSwapRouter.changeOperatingFee(uint256) (contracts/swap/rAskoSwapRouter.sol#87-93)
changeBuybackFee(uint256) should be declared external:
 - rAskoSwapRouter.changeBuybackFee(uint256) (contracts/swap/rAskoSwapRouter.sol#95-101)
changeBaseLR(address) should be declared external:
 - rAskoSwapRouter.changeBaseLR(address) (contracts/swap/rAskoSwapRouter.sol#103-105)
```

```
changeBaseHR(address) should be declared external:
 - rAskoSwapRouter.changeBaseHR(address) (contracts/swap/rAskoSwapRouter.sol#107-109)
changeDaoAddress(address) should be declared external:
 - rAskoSwapRouter.changeDaoAddress(address) (contracts/swap/rAskoSwapRouter.sol#111-113)
changeOperating(address) should be declared external:
 - rAskoSwapRouter.changeOperating(address) (contracts/swap/rAskoSwapRouter.sol#115-117)
changeBuybackAddress(address) should be declared external:
 - rAskoSwapRouter.changeBuybackAddress(address) (contracts/swap/
rAskoSwapRouter.sol#119-121)
checkPairFor(address,address) should be declared external:
 - rAskoSwapRouter.checkPairFor(address,address) (contracts/swap/
rAskoSwapRouter.sol#123-129)
quote(uint256,uint256,uint256) should be declared external:
 - rAskoSwapRouter.quote(uint256,uint256,uint256) (contracts/swap/
rAskoSwapRouter.sol#445-451)
getAmountOut(uint256,uint256,uint256) should be declared external:
 - rAskoSwapRouter.getAmountOut(uint256,uint256,uint256) (contracts/swap/
rAskoSwapRouter.sol#453-465)
getAmountIn(uint256,uint256,uint256) should be declared external:
 - rAskoSwapRouter.getAmountIn(uint256,uint256,uint256) (contracts/swap/
rAskoSwapRouter.sol#467-479)
getAmountsOut(uint256,address[]) should be declared external:
 - rAskoSwapRouter.getAmountsOut(uint256,address[]) (contracts/swap/
rAskoSwapRouter.sol#481-487)
getAmountsIn(uint256,address[]) should be declared external:
 - rAskoSwapRouter.getAmountsIn(uint256,address[]) (contracts/swap/
rAskoSwapRouter.sol#489-495)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-
that-could-be-declared-external

owner() should be declared external:
 - Ownable.owner() (node_modules/@pancakeswap/pancake-swap-lib/contracts/access/
Ownable.sol#36-38)
renounceOwnership() should be declared external:
 - Ownable.renounceOwnership() (node_modules/@pancakeswap/pancake-swap-lib/contracts/
access/Ownable.sol#55-58)
transferOwnership(address) should be declared external:
 - Ownable.transferOwnership(address) (node_modules/@pancakeswap/pancake-swap-lib/
contracts/access/Ownable.sol#64-66)
updateMultiplier(uint256) should be declared external:
 - rAskoFarm.updateMultiplier(uint256) (contracts/farm/rAskoFarm.sol#101-103)
changeApyCalculator(address) should be declared external:
 - rAskoFarm.changeApyCalculator(address) (contracts/farm/rAskoFarm.sol#105-107)
```

```
add(uint256,IBEP20,bool) should be declared external:
 - rAskoFarm.add(uint256,IBEP20,bool) (contracts/farm/rAskoFarm.sol#115-128)
set(uint256,uint256,bool) should be declared external:
 - rAskoFarm.set(uint256,uint256,bool) (contracts/farm/rAskoFarm.sol#131-141)
setMigrator(IMigratorChef) should be declared external:
 - rAskoFarm.setMigrator(IMigratorChef) (contracts/farm/rAskoFarm.sol#157-159)
migrate(uint256) should be declared external:
 - rAskoFarm.migrate(uint256) (contracts/farm/rAskoFarm.sol#162-171)
deposit(uint256,uint256) should be declared external:
 - rAskoFarm.deposit(uint256,uint256) (contracts/farm/rAskoFarm.sol#231-252)
withdraw(uint256,uint256) should be declared external:
 - rAskoFarm.withdraw(uint256,uint256) (contracts/farm/rAskoFarm.sol#255-282)
emergencyWithdraw(uint256) should be declared external:
 - rAskoFarm.emergencyWithdraw(uint256) (contracts/farm/rAskoFarm.sol#285-292)
dev(address) should be declared external:
 - rAskoFarm.dev(address) (contracts/farm/rAskoFarm.sol#295-298)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-
that-could-be-declared-external

grantRole(bytes32,address) should be declared external:
 - AccessControl.grantRole(bytes32,address) (node_modules/@openzeppelin/contracts/access/
AccessControl.sol#163-165)
revokeRole(bytes32,address) should be declared external:
 - AccessControl.revokeRole(bytes32,address) (node_modules/@openzeppelin/contracts/access/
AccessControl.sol#176-178)
renounceRole(bytes32,address) should be declared external:
 - AccessControl.renounceRole(bytes32,address) (node_modules/@openzeppelin/contracts/
access/AccessControl.sol#194-198)
name() should be declared external:
 - ERC20.name() (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#60-62)
symbol() should be declared external:
 - ERC20.symbol() (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#68-70)
totalSupply() should be declared external:
 - ERC20.totalSupply() (node_modules/@openzeppelin/contracts/token/ERC20/ERC20.sol#92-94)
balanceOf(address) should be declared external:
 - ERC20.balanceOf(address) (node_modules/@openzeppelin/contracts/token/ERC20/
ERC20.sol#99-101)
transfer(address,uint256) should be declared external:
 - ERC20.transfer(address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/
ERC20.sol#111-114)
allowance(address,address) should be declared external:
 - ERC20.allowance(address,address) (node_modules/@openzeppelin/contracts/token/ERC20/
ERC20.sol#119-121)
```

```
approve(address,uint256) should be declared external:
 - ERC20.approve(address,uint256) (node_modules/@openzeppelin/contracts/token/ERC20/
ERC20.sol#130-133)
transferFrom(address,address,uint256) should be declared external:
 - ERC20.transferFrom(address,address,uint256) (node_modules/@openzeppelin/contracts/token/
ERC20/ERC20.sol#148-156)
increaseAllowance(address,uint256) should be declared external:
 - ERC20.increaseAllowance(address,uint256) (node_modules/@openzeppelin/contracts/token/
ERC20/ERC20.sol#170-173)
decreaseAllowance(address,uint256) should be declared external:
 - ERC20.decreaseAllowance(address,uint256) (node_modules/@openzeppelin/contracts/token/
ERC20/ERC20.sol#189-195)
valueOf3000(address) should be declared external:
 - APYCalculator.valueOf3000(address) (contracts/farm/APYCalculator.sol#23-25)
addAdmin(address) should be declared external:
 - APYCalculator.addAdmin(address) (contracts/farm/APYCalculator.sol#27-29)
changeValues(address[],uint256[]) should be declared external:
 - APYCalculator.changeValues(address[],uint256[]) (contracts/farm/
APYCalculator.sol#31-36)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-
that-could-be-declared-external
. analyzed (57 contracts with 75 detectors), 234 result(s) found
```

contact@hashex.org

@hashex_manager

blog.hashex.org

linkedin

github

twitter

HashEx

BLOCKCHAIN SECURITY