PathFund Smart Contract Final Audit Report

Scope of Audit

The scope of this audit was to analyse and document the PathFund Staking Token smart contract codebase for quality, security, and correctness.

Check Vulnerabilities

- Re-entrancy
- Timestamp Dependence
- Gas Limit and Loops
- DoS with Block Gas Limit
- Transaction-Ordering Dependence
- Use of tx.origin
- Exception disorder
- Gasless send
- Balance equality
- Byte array
- Transfer forwards all gas
- ERC20 API violation
- Malicious libraries
- Compiler version not fixed
- Redundant fallback function
- Send instead of transfer
- Style guide violation
- Unchecked external call
- Unchecked maths
- Unsafe type inference
- Implicit visibility level

Techniques and Methods

Throughout the audit of smart contracts, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments match logic and expected behaviour.
- Token distribution and calculations are as per the intended behaviour mentioned in the whitepaper.
- Implementation of ERC-20 token standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods, and tools were used to review all the smart contracts.

Structural Analysis

In this step, we have analyzed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.

Static Analysis

A static Analysis of Smart Contracts was done to identify contract vulnerabilities. In this step, a series of automated tools are used to test the security of smart contracts.

Code Review / Manual Analysis

Manual Analysis or review of code was done to identify new vulnerabilities or verify the vulnerabilities found during the static analysis. Contracts were completely manually analyzed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of the automated analysis were manually verified.

Gas Consumption

In this step, we have checked the behaviour of smart contracts in production. Checks were done to know how much gas gets consumed and the possibilities of optimization of code to reduce gas consumption.

Tools and Platforms used for Audit

Remix IDE, Truffle, Truffle Team, Solhint, Mythril, Slither, Solidity statistic analysis.

Issue Categories

Every issue in this report has been assigned to a severity level. There are four levels of severity, and each of them has been explained below.

High Severity Issues

A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract’s performance or functionality, and we recommend these issues be fixed before moving to a live environment.

Medium Severity Issues

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems, and they should still be fixed.

Low Severity Issues

Low-level severity issues can cause minor impact and or are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.

Informational Issues

These are four severity issues that indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

Number of security issues per severity.

TYPE	HIGH	MEDIUM	LOW	INFORMATIONAL
Open	0	0	0	3
Acknowledged	0	2	2	0
Closed	0	0	0	0

Introduction

During the period of **March 11, 2021 to March 19, 2021** - QuillAudits Team performed a security audit for **PathFund** smart contract.

The code for the audit was taken from following the official link**:**

Codebase: Shared a zip file of the smart contract.

Contract - PathFund

Issues Found – Code Review / Manual Testing

High Severity Issues

NONE

Medium Severity Issues

- **[M1] kkAirDrop function for loop upper bond check missing**

  The function for loop should be checked for the upper threshold of address that can be incorporated in one single transaction and the for loop should run for that

  Status: Acknowledge

- **[M2] Missing check for address and amount array**

  The check for input length of address and amount should be equal in kkAirDrop.

  Status: Acknowledge

Low Severity Issues

- **[L1] Remove unused function isThisFrom_KKteam**

  IsThisFrom function is not been used anywhere we recommend to remove the function.

  Status: Acknowledge

- **[L2] Zero check's missing**

  zero check lacks in setairDropAddress, updateUniswapV2Router, setFeesAddress arguments

Status: Acknowledge

Informational Issues

- **[INF1] public functions should be set to external**

  setFeesAddress, setairDropAddress, retunList_excludedFee, retunList_premarket, retunList_blacklist, should be external from private.

  ‘public’ functions that are never used within the contract should be declared ‘external’ to save gas.

  Reference

  Status: Open

- **[INF2] use uint256 instead of uint**

  By using uint256 the readability of the code get improved and user is able to identify the bytes length of it integer.

  Status: Open

- **[INF3] emit events for state change.**

  Whenever there is a state change we recommend to emit the same.

  switchBlockMultiBuys, setFee, setSwapAndLiquify, set_maxLimits should emit event at the change of state.

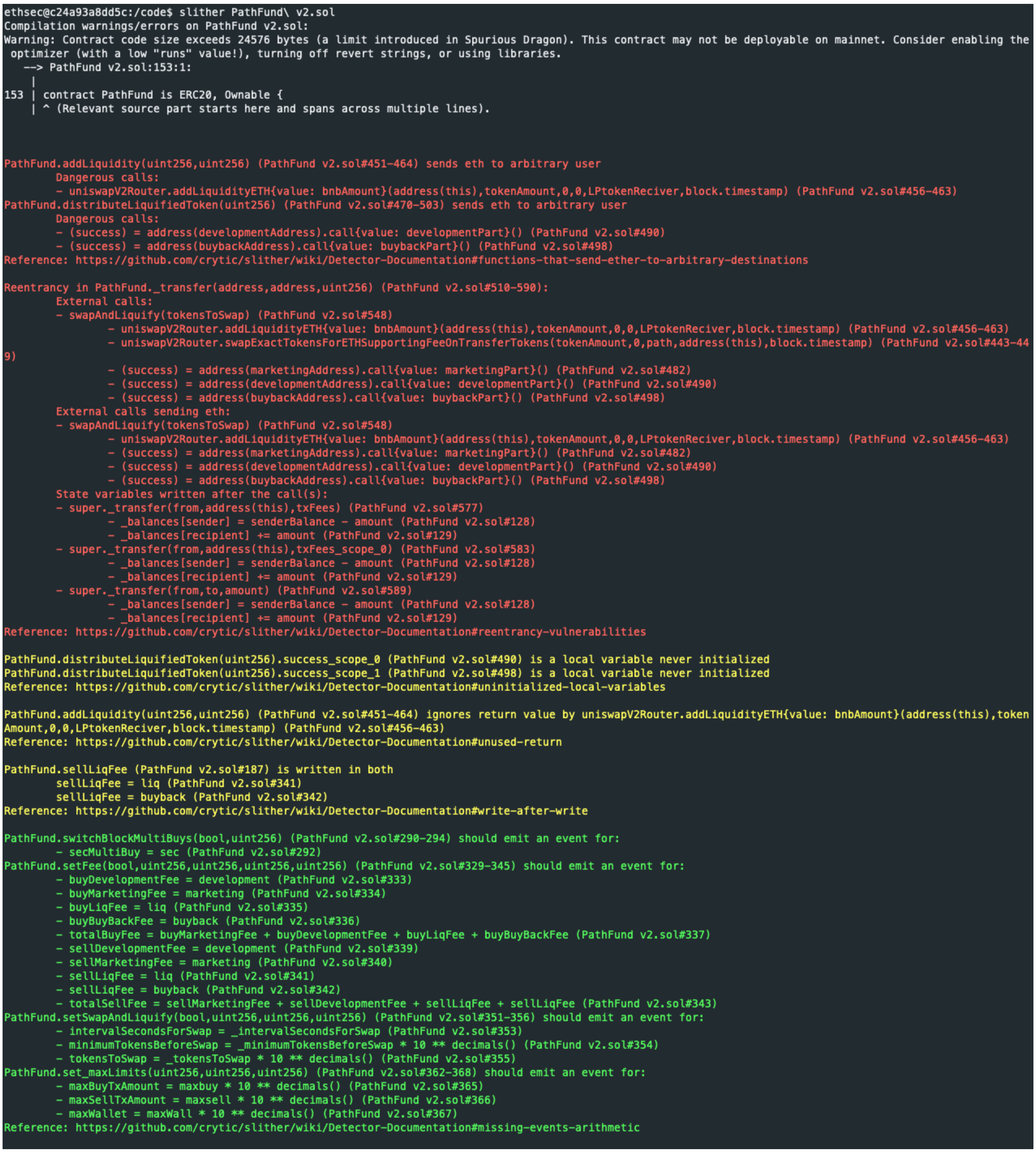
  Status: Open

Binance Testnet Contract

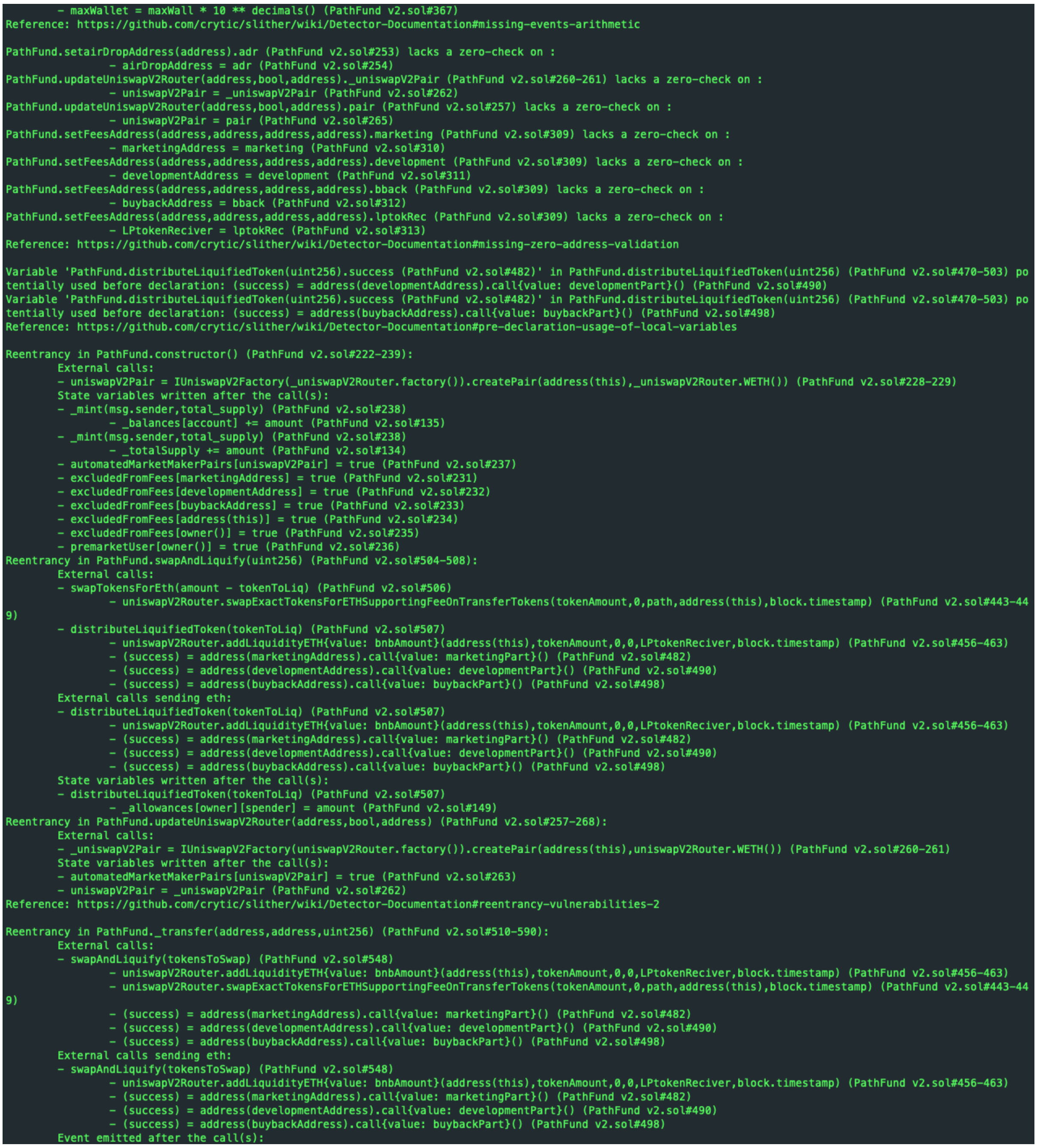
0x45c66f7536d6233297E96C7c8fB9238AdE0B1b9d

Automated Tests

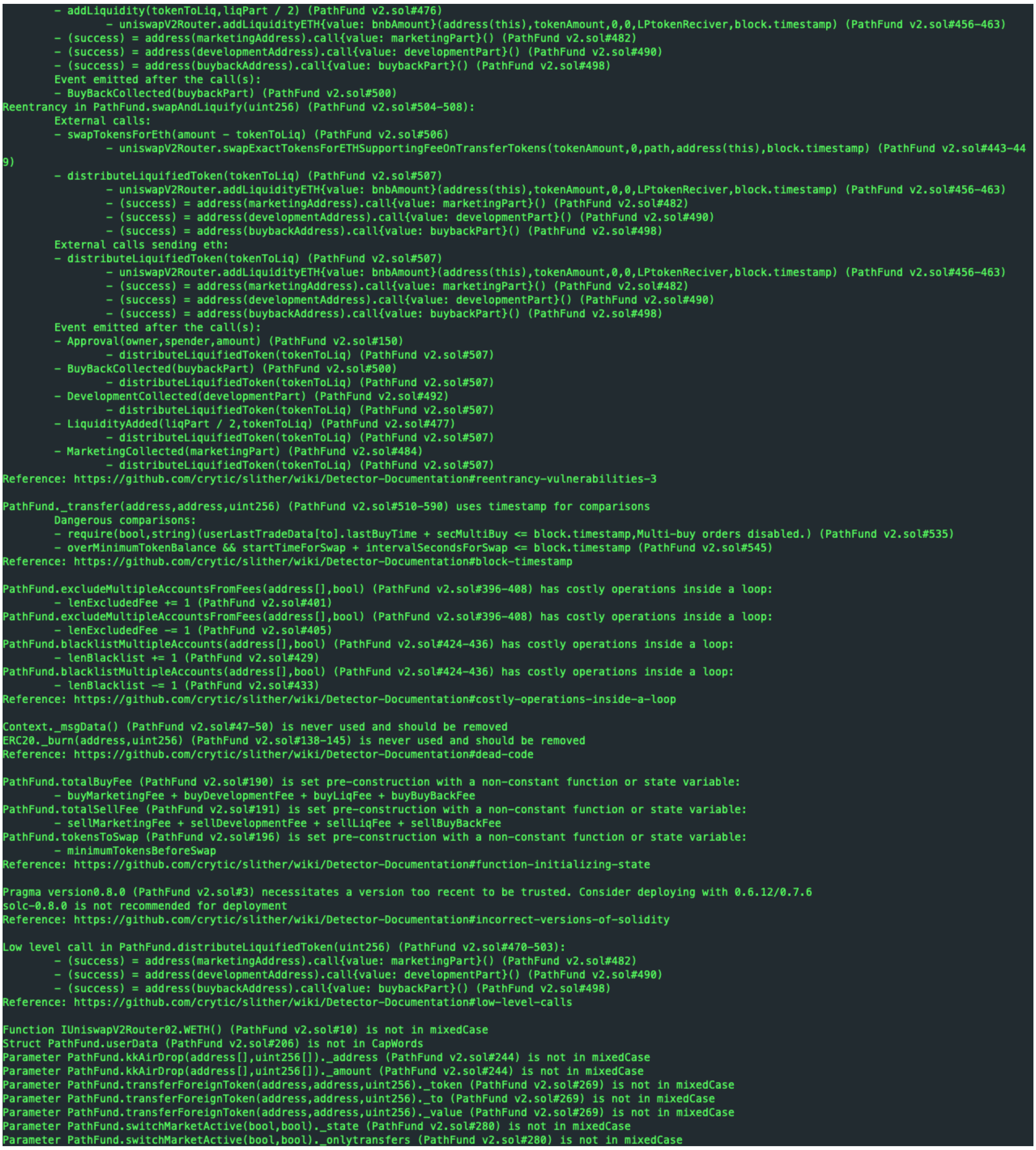
Slither:



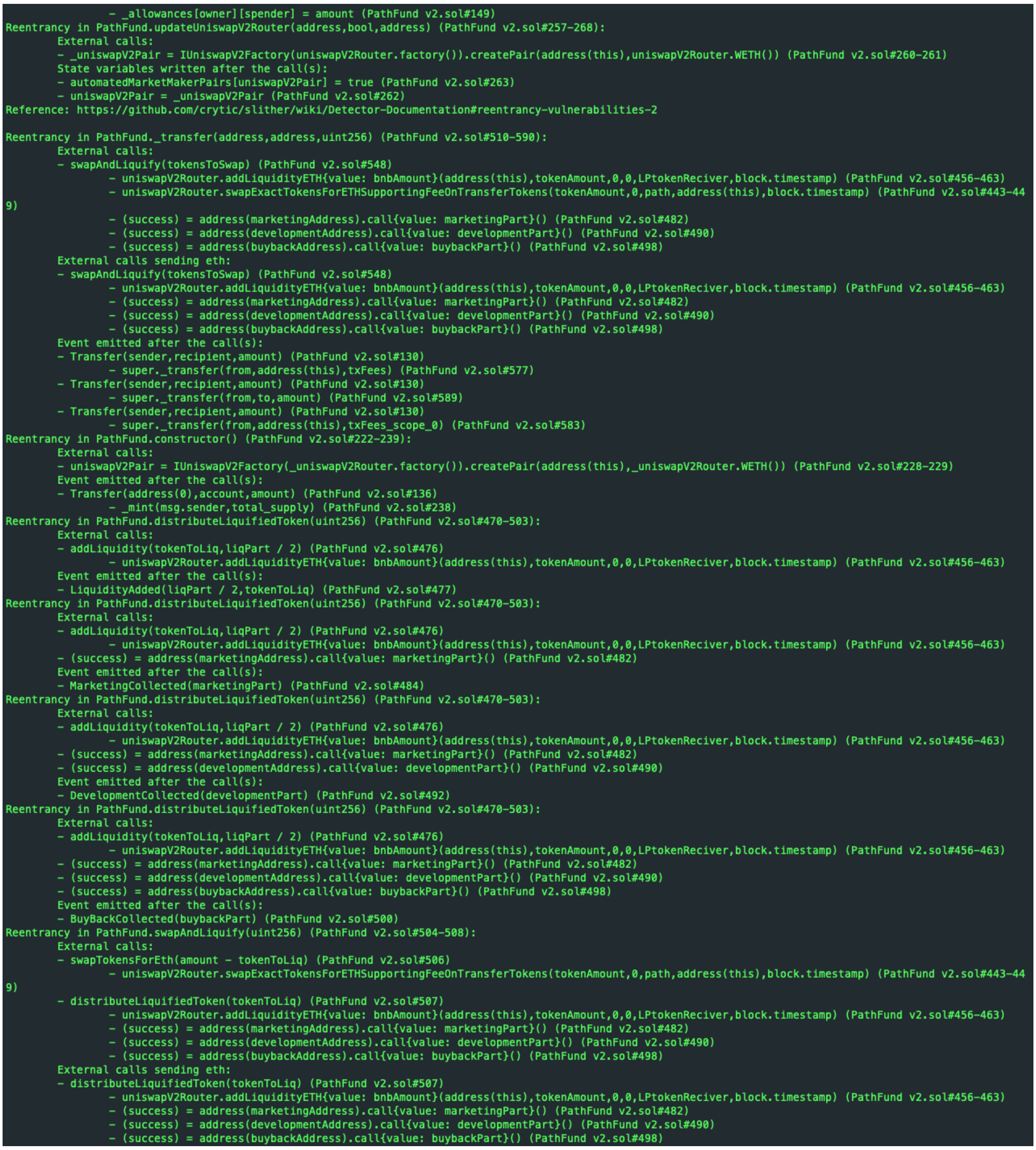
```
ethsec@c24a93a8dd5c:/code$ slither PathFund\ v2.sol
Compilation warnings/errors on PathFund v2.sol:
Warning: Contract code size exceeds 24576 bytes (a limit introduced in Spurious Dragon). This contract may not be deployable on mainnet. Consider enabling the
 optimizer (with a low "runs" value!), turning off revert strings, or using libraries.
   --> PathFund v2.sol:153:1:
    |
153 | contract PathFund is ERC20, Ownable {
    | ^ (Relevant source part starts here and spans across multiple lines).



PathFund.addLiquidity(uint256,uint256) (PathFund v2.sol#451-464) sends eth to arbitrary user
        Dangerous calls:
        - uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),tokenAmount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
PathFund.distributeLiquifiedToken(uint256) (PathFund v2.sol#470-503) sends eth to arbitrary user
        Dangerous calls:
        - (success) = address(developmentAddress).call{value: developmentPart}() (PathFund v2.sol#490)
        - (success) = address(buybackAddress).call{value: buybackPart}() (PathFund v2.sol#498)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#functions-that-send-ether-to-arbitrary-destinations

Reentrancy in PathFund._transfer(address,address,uint256) (PathFund v2.sol#510-590):
        External calls:
        - swapAndLiquify(tokensToSwap) (PathFund v2.sol#548)
                - uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),tokenAmount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (PathFund v2.sol#443-44
9)
                - (success) = address(marketingAddress).call{value: marketingPart}() (PathFund v2.sol#482)
                - (success) = address(developmentAddress).call{value: developmentPart}() (PathFund v2.sol#490)
                - (success) = address(buybackAddress).call{value: buybackPart}() (PathFund v2.sol#498)
        External calls sending eth:
        - swapAndLiquify(tokensToSwap) (PathFund v2.sol#548)
                - uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),tokenAmount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
                - (success) = address(marketingAddress).call{value: marketingPart}() (PathFund v2.sol#482)
                - (success) = address(developmentAddress).call{value: developmentPart}() (PathFund v2.sol#490)
                - (success) = address(buybackAddress).call{value: buybackPart}() (PathFund v2.sol#498)
        State variables written after the call(s):
        - super._transfer(from,address(this),txFees) (PathFund v2.sol#577)
                - _balances[sender] = senderBalance - amount (PathFund v2.sol#128)
                - _balances[recipient] += amount (PathFund v2.sol#129)
        - super._transfer(from,address(this),txFees_scope_0) (PathFund v2.sol#583)
                - _balances[sender] = senderBalance - amount (PathFund v2.sol#128)
                - _balances[recipient] += amount (PathFund v2.sol#129)
        - super._transfer(from,to,amount) (PathFund v2.sol#589)
                - _balances[sender] = senderBalance - amount (PathFund v2.sol#128)
                - _balances[recipient] += amount (PathFund v2.sol#129)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities

PathFund.distributeLiquifiedToken(uint256).success_scope_0 (PathFund v2.sol#490) is a local variable never initialized
PathFund.distributeLiquifiedToken(uint256).success_scope_1 (PathFund v2.sol#498) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables

PathFund.addLiquidity(uint256,uint256) (PathFund v2.sol#451-464) ignores return value by uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),token
Amount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

PathFund.sellLiqFee (PathFund v2.sol#187) is written in both
        sellLiqFee = liq (PathFund v2.sol#341)
        sellLiqFee = buyback (PathFund v2.sol#342)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#write-after-write

PathFund.switchBlockMultiBuys(bool,uint256) (PathFund v2.sol#290-294) should emit an event for:
        - secMultiBuy = sec (PathFund v2.sol#292)
PathFund.setFee(bool,uint256,uint256,uint256,uint256) (PathFund v2.sol#329-345) should emit an event for:
        - buyDevelopmentFee = development (PathFund v2.sol#333)
        - buyMarketingFee = marketing (PathFund v2.sol#334)
        - buyLiqFee = liq (PathFund v2.sol#335)
        - buyBuyBackFee = buyback (PathFund v2.sol#336)
        - totalBuyFee = buyMarketingFee + buyDevelopmentFee + buyLiqFee + buyBuyBackFee (PathFund v2.sol#337)
        - sellDevelopmentFee = development (PathFund v2.sol#339)
        - sellMarketingFee = marketing (PathFund v2.sol#340)
        - sellLiqFee = liq (PathFund v2.sol#341)
        - sellLiqFee = buyback (PathFund v2.sol#342)
        - totalSellFee = sellMarketingFee + sellDevelopmentFee + sellLiqFee + sellLiqFee (PathFund v2.sol#343)
PathFund.setSwapAndLiquify(bool,uint256,uint256,uint256) (PathFund v2.sol#351-356) should emit an event for:
        - intervalSecondsForSwap = _intervalSecondsForSwap (PathFund v2.sol#353)
        - minimumTokensBeforeSwap = _minimumTokensBeforeSwap * 10 ** decimals() (PathFund v2.sol#354)
        - tokensToSwap = _tokensToSwap * 10 ** decimals() (PathFund v2.sol#355)
PathFund.set_maxLimits(uint256,uint256,uint256) (PathFund v2.sol#362-368) should emit an event for:
        - maxBuyTxAmount = maxbuy * 10 ** decimals() (PathFund v2.sol#365)
        - maxSellTxAmount = maxsell * 10 ** decimals() (PathFund v2.sol#366)
        - maxWallet = maxWall * 10 ** decimals() (PathFund v2.sol#367)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic
```



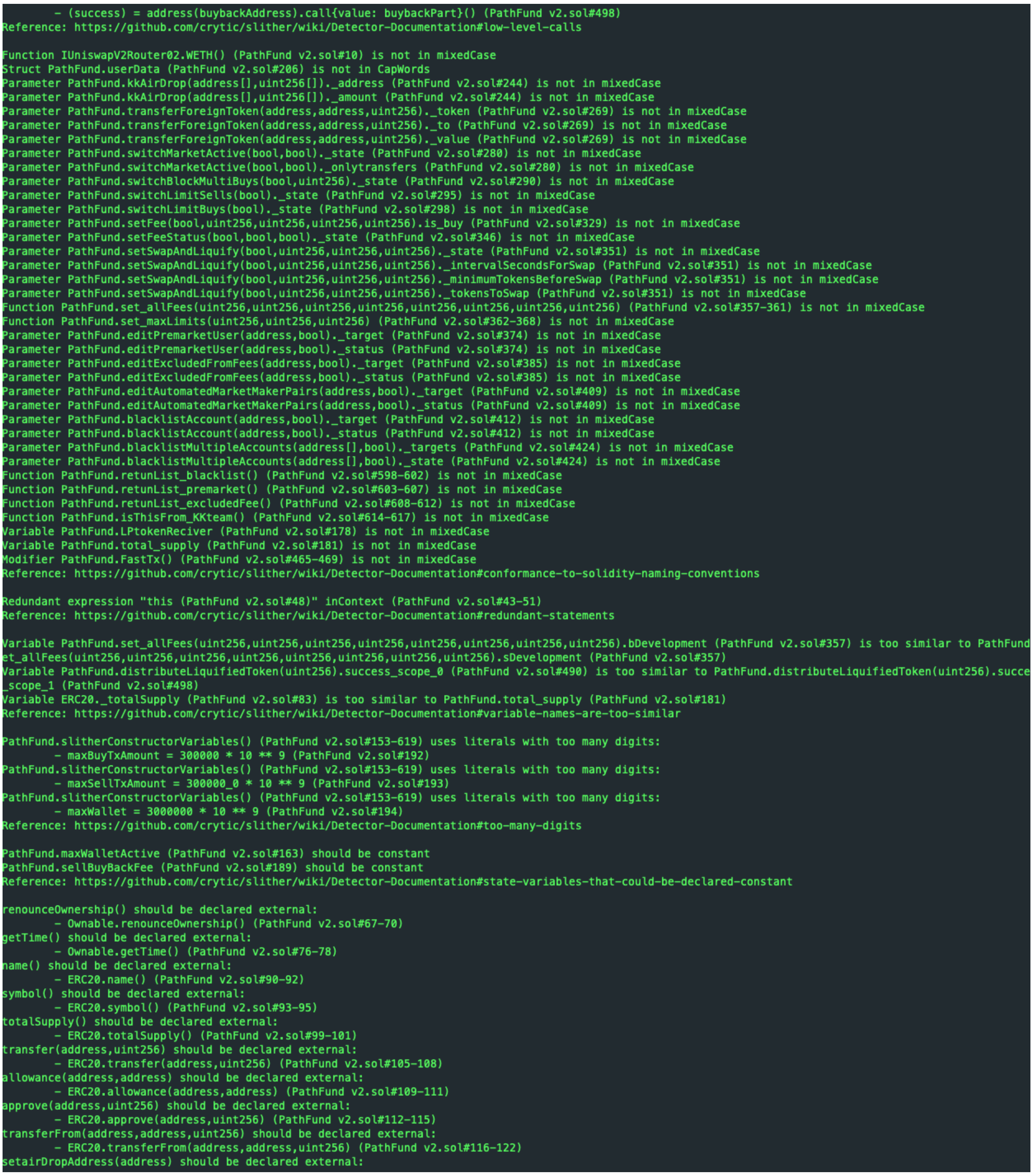
```
        - maxWallet = maxWall * 10 ** decimals() (PathFund v2.sol#367)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic

PathFund.setairDropAddress(address).adr (PathFund v2.sol#253) lacks a zero-check on :
                - airDropAddress = adr (PathFund v2.sol#254)
PathFund.updateUniswapV2Router(address,bool,address)._uniswapV2Pair (PathFund v2.sol#260-261) lacks a zero-check on :
                - uniswapV2Pair = _uniswapV2Pair (PathFund v2.sol#262)
PathFund.updateUniswapV2Router(address,bool,address).pair (PathFund v2.sol#257) lacks a zero-check on :
                - uniswapV2Pair = pair (PathFund v2.sol#265)
PathFund.setFeesAddress(address,address,address,address).marketing (PathFund v2.sol#309) lacks a zero-check on :
                - marketingAddress = marketing (PathFund v2.sol#310)
PathFund.setFeesAddress(address,address,address,address).development (PathFund v2.sol#309) lacks a zero-check on :
                - developmentAddress = development (PathFund v2.sol#311)
PathFund.setFeesAddress(address,address,address,address).bback (PathFund v2.sol#309) lacks a zero-check on :
                - buybackAddress = bback (PathFund v2.sol#312)
PathFund.setFeesAddress(address,address,address,address).lptokRec (PathFund v2.sol#309) lacks a zero-check on :
                - LPtokenReciver = lptokRec (PathFund v2.sol#313)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

Variable 'PathFund.distributeLiquifiedToken(uint256).success (PathFund v2.sol#482)' in PathFund.distributeLiquifiedToken(uint256) (PathFund v2.sol#470-503) po
tentially used before declaration: (success) = address(developmentAddress).call{value: developmentPart}() (PathFund v2.sol#490)
Variable 'PathFund.distributeLiquifiedToken(uint256).success (PathFund v2.sol#482)' in PathFund.distributeLiquifiedToken(uint256) (PathFund v2.sol#470-503) po
tentially used before declaration: (success) = address(buybackAddress).call{value: buybackPart}() (PathFund v2.sol#498)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-declaration-usage-of-local-variables

Reentrancy in PathFund.constructor() (PathFund v2.sol#222-239):
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (PathFund v2.sol#228-229)
        State variables written after the call(s):
        - _mint(msg.sender,total_supply) (PathFund v2.sol#238)
                - _balances[account] += amount (PathFund v2.sol#135)
        - _mint(msg.sender,total_supply) (PathFund v2.sol#238)
                - _totalSupply += amount (PathFund v2.sol#134)
        - automatedMarketMakerPairs[uniswapV2Pair] = true (PathFund v2.sol#237)
        - excludedFromFees[marketingAddress] = true (PathFund v2.sol#231)
        - excludedFromFees[developmentAddress] = true (PathFund v2.sol#232)
        - excludedFromFees[buybackAddress] = true (PathFund v2.sol#233)
        - excludedFromFees[address(this)] = true (PathFund v2.sol#234)
        - excludedFromFees[owner()] = true (PathFund v2.sol#235)
        - premarketUser[owner()] = true (PathFund v2.sol#236)
Reentrancy in PathFund.swapAndLiquify(uint256) (PathFund v2.sol#504-508):
        External calls:
        - swapTokensForEth(amount - tokenToLiq) (PathFund v2.sol#506)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (PathFund v2.sol#443-44
9)
        - distributeLiquifiedToken(tokenToLiq) (PathFund v2.sol#507)
                - uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),tokenAmount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
                - (success) = address(marketingAddress).call{value: marketingPart}() (PathFund v2.sol#482)
                - (success) = address(developmentAddress).call{value: developmentPart}() (PathFund v2.sol#490)
                - (success) = address(buybackAddress).call{value: buybackPart}() (PathFund v2.sol#498)
        External calls sending eth:
        - distributeLiquifiedToken(tokenToLiq) (PathFund v2.sol#507)
                - uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),tokenAmount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
                - (success) = address(marketingAddress).call{value: marketingPart}() (PathFund v2.sol#482)
                - (success) = address(developmentAddress).call{value: developmentPart}() (PathFund v2.sol#490)
                - (success) = address(buybackAddress).call{value: buybackPart}() (PathFund v2.sol#498)
        State variables written after the call(s):
        - distributeLiquifiedToken(tokenToLiq) (PathFund v2.sol#507)
                - _allowances[owner][spender] = amount (PathFund v2.sol#149)
Reentrancy in PathFund.updateUniswapV2Router(address,bool,address) (PathFund v2.sol#257-268):
        External calls:
        - _uniswapV2Pair = IUniswapV2Factory(uniswapV2Router.factory()).createPair(address(this),uniswapV2Router.WETH()) (PathFund v2.sol#260-261)
        State variables written after the call(s):
        - automatedMarketMakerPairs[uniswapV2Pair] = true (PathFund v2.sol#263)
        - uniswapV2Pair = _uniswapV2Pair (PathFund v2.sol#262)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2

Reentrancy in PathFund._transfer(address,address,uint256) (PathFund v2.sol#510-590):
        External calls:
        - swapAndLiquify(tokensToSwap) (PathFund v2.sol#548)
                - uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),tokenAmount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (PathFund v2.sol#443-44
9)
                - (success) = address(marketingAddress).call{value: marketingPart}() (PathFund v2.sol#482)
                - (success) = address(developmentAddress).call{value: developmentPart}() (PathFund v2.sol#490)
                - (success) = address(buybackAddress).call{value: buybackPart}() (PathFund v2.sol#498)
        External calls sending eth:
        - swapAndLiquify(tokensToSwap) (PathFund v2.sol#548)
                - uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),tokenAmount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
                - (success) = address(marketingAddress).call{value: marketingPart}() (PathFund v2.sol#482)
                - (success) = address(developmentAddress).call{value: developmentPart}() (PathFund v2.sol#490)
                - (success) = address(buybackAddress).call{value: buybackPart}() (PathFund v2.sol#498)
        Event emitted after the call(s):
```



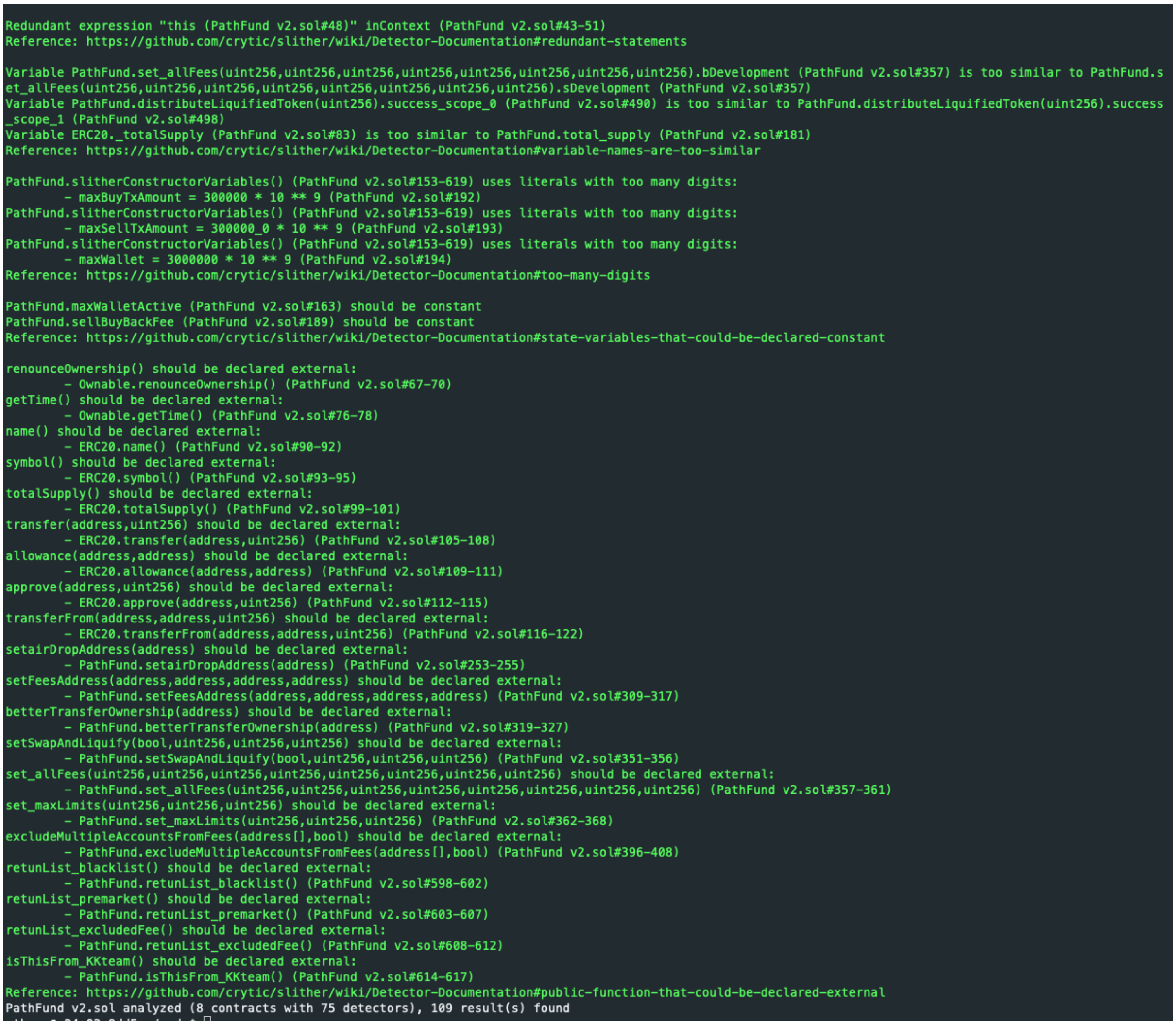
```
        - addLiquidity(tokenToLiq,liqPart / 2) (PathFund v2.sol#476)
                - uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),tokenAmount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
        - (success) = address(marketingAddress).call{value: marketingPart}() (PathFund v2.sol#482)
        - (success) = address(developmentAddress).call{value: developmentPart}() (PathFund v2.sol#490)
        - (success) = address(buybackAddress).call{value: buybackPart}() (PathFund v2.sol#498)
        Event emitted after the call(s):
        - BuyBackCollected(buybackPart) (PathFund v2.sol#500)
Reentrancy in PathFund.swapAndLiquify(uint256) (PathFund v2.sol#504-508):
        External calls:
        - swapTokensForEth(amount - tokenToLiq) (PathFund v2.sol#506)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (PathFund v2.sol#443-44
9)
        - distributeLiquifiedToken(tokenToLiq) (PathFund v2.sol#507)
                - uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),tokenAmount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
                - (success) = address(marketingAddress).call{value: marketingPart}() (PathFund v2.sol#482)
                - (success) = address(developmentAddress).call{value: developmentPart}() (PathFund v2.sol#490)
                - (success) = address(buybackAddress).call{value: buybackPart}() (PathFund v2.sol#498)
        External calls sending eth:
        - distributeLiquifiedToken(tokenToLiq) (PathFund v2.sol#507)
                - uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),tokenAmount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
                - (success) = address(marketingAddress).call{value: marketingPart}() (PathFund v2.sol#482)
                - (success) = address(developmentAddress).call{value: developmentPart}() (PathFund v2.sol#490)
                - (success) = address(buybackAddress).call{value: buybackPart}() (PathFund v2.sol#498)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (PathFund v2.sol#150)
                - distributeLiquifiedToken(tokenToLiq) (PathFund v2.sol#507)
        - BuyBackCollected(buybackPart) (PathFund v2.sol#500)
                - distributeLiquifiedToken(tokenToLiq) (PathFund v2.sol#507)
        - DevelopmentCollected(developmentPart) (PathFund v2.sol#492)
                - distributeLiquifiedToken(tokenToLiq) (PathFund v2.sol#507)
        - LiquidityAdded(liqPart / 2,tokenToLiq) (PathFund v2.sol#477)
                - distributeLiquifiedToken(tokenToLiq) (PathFund v2.sol#507)
        - MarketingCollected(marketingPart) (PathFund v2.sol#484)
                - distributeLiquifiedToken(tokenToLiq) (PathFund v2.sol#507)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

PathFund._transfer(address,address,uint256) (PathFund v2.sol#510-590) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(userLastTradeData[to].lastBuyTime + secMultiBuy <= block.timestamp,Multi-buy orders disabled.) (PathFund v2.sol#535)
        - overMinimumTokenBalance && startTimeForSwap + intervalSecondsForSwap <= block.timestamp (PathFund v2.sol#545)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

PathFund.excludeMultipleAccountsFromFees(address[],bool) (PathFund v2.sol#396-408) has costly operations inside a loop:
        - lenExcludedFee += 1 (PathFund v2.sol#401)
PathFund.excludeMultipleAccountsFromFees(address[],bool) (PathFund v2.sol#396-408) has costly operations inside a loop:
        - lenExcludedFee -= 1 (PathFund v2.sol#405)
PathFund.blacklistMultipleAccounts(address[],bool) (PathFund v2.sol#424-436) has costly operations inside a loop:
        - lenBlacklist += 1 (PathFund v2.sol#429)
PathFund.blacklistMultipleAccounts(address[],bool) (PathFund v2.sol#424-436) has costly operations inside a loop:
        - lenBlacklist -= 1 (PathFund v2.sol#433)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop

Context._msgData() (PathFund v2.sol#47-50) is never used and should be removed
ERC20._burn(address,uint256) (PathFund v2.sol#138-145) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

PathFund.totalBuyFee (PathFund v2.sol#190) is set pre-construction with a non-constant function or state variable:
        - buyMarketingFee + buyDevelopmentFee + buyLiqFee + buyBuyBackFee
PathFund.totalSellFee (PathFund v2.sol#191) is set pre-construction with a non-constant function or state variable:
        - sellMarketingFee + sellDevelopmentFee + sellLiqFee + sellBuyBackFee
PathFund.tokensToSwap (PathFund v2.sol#196) is set pre-construction with a non-constant function or state variable:
        - minimumTokensBeforeSwap
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#function-initializing-state

Pragma version0.8.0 (PathFund v2.sol#3) 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 PathFund.distributeLiquifiedToken(uint256) (PathFund v2.sol#470-503):
        - (success) = address(marketingAddress).call{value: marketingPart}() (PathFund v2.sol#482)
        - (success) = address(developmentAddress).call{value: developmentPart}() (PathFund v2.sol#490)
        - (success) = address(buybackAddress).call{value: buybackPart}() (PathFund v2.sol#498)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function IUniswapV2Router02.WETH() (PathFund v2.sol#10) is not in mixedCase
Struct PathFund.userData (PathFund v2.sol#206) is not in CapWords
Parameter PathFund.kkAirDrop(address[],uint256[])._address (PathFund v2.sol#244) is not in mixedCase
Parameter PathFund.kkAirDrop(address[],uint256[])._amount (PathFund v2.sol#244) is not in mixedCase
Parameter PathFund.transferForeignToken(address,address,uint256)._token (PathFund v2.sol#269) is not in mixedCase
Parameter PathFund.transferForeignToken(address,address,uint256)._to (PathFund v2.sol#269) is not in mixedCase
Parameter PathFund.transferForeignToken(address,address,uint256)._value (PathFund v2.sol#269) is not in mixedCase
Parameter PathFund.switchMarketActive(bool,bool)._state (PathFund v2.sol#280) is not in mixedCase
Parameter PathFund.switchMarketActive(bool,bool)._onlytransfers (PathFund v2.sol#280) is not in mixedCase
```



```
                - _allowances[owner][spender] = amount (PathFund v2.sol#149)
Reentrancy in PathFund.updateUniswapV2Router(address,bool,address) (PathFund v2.sol#257-268):
        External calls:
        - _uniswapV2Pair = IUniswapV2Factory(uniswapV2Router.factory()).createPair(address(this),uniswapV2Router.WETH()) (PathFund v2.sol#260-261)
        State variables written after the call(s):
        - automatedMarketMakerPairs[uniswapV2Pair] = true (PathFund v2.sol#263)
        - uniswapV2Pair = _uniswapV2Pair (PathFund v2.sol#262)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2

Reentrancy in PathFund._transfer(address,address,uint256) (PathFund v2.sol#510-590):
        External calls:
        - swapAndLiquify(tokensToSwap) (PathFund v2.sol#548)
                - uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),tokenAmount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (PathFund v2.sol#443-44
9)
                - (success) = address(marketingAddress).call{value: marketingPart}() (PathFund v2.sol#482)
                - (success) = address(developmentAddress).call{value: developmentPart}() (PathFund v2.sol#490)
                - (success) = address(buybackAddress).call{value: buybackPart}() (PathFund v2.sol#498)
        External calls sending eth:
        - swapAndLiquify(tokensToSwap) (PathFund v2.sol#548)
                - uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),tokenAmount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
                - (success) = address(marketingAddress).call{value: marketingPart}() (PathFund v2.sol#482)
                - (success) = address(developmentAddress).call{value: developmentPart}() (PathFund v2.sol#490)
                - (success) = address(buybackAddress).call{value: buybackPart}() (PathFund v2.sol#498)
        Event emitted after the call(s):
        - Transfer(sender,recipient,amount) (PathFund v2.sol#130)
                - super._transfer(from,address(this),txFees) (PathFund v2.sol#577)
        - Transfer(sender,recipient,amount) (PathFund v2.sol#130)
                - super._transfer(from,to,amount) (PathFund v2.sol#589)
        - Transfer(sender,recipient,amount) (PathFund v2.sol#130)
                - super._transfer(from,address(this),txFees_scope_0) (PathFund v2.sol#583)
Reentrancy in PathFund.constructor() (PathFund v2.sol#222-239):
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (PathFund v2.sol#228-229)
        Event emitted after the call(s):
        - Transfer(address(0),account,amount) (PathFund v2.sol#136)
                - _mint(msg.sender,total_supply) (PathFund v2.sol#238)
Reentrancy in PathFund.distributeLiquifiedToken(uint256) (PathFund v2.sol#470-503):
        External calls:
        - addLiquidity(tokenToLiq,liqPart / 2) (PathFund v2.sol#476)
                - uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),tokenAmount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
        Event emitted after the call(s):
        - LiquidityAdded(liqPart / 2,tokenToLiq) (PathFund v2.sol#477)
Reentrancy in PathFund.distributeLiquifiedToken(uint256) (PathFund v2.sol#470-503):
        External calls:
        - addLiquidity(tokenToLiq,liqPart / 2) (PathFund v2.sol#476)
                - uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),tokenAmount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
        - (success) = address(marketingAddress).call{value: marketingPart}() (PathFund v2.sol#482)
        Event emitted after the call(s):
        - MarketingCollected(marketingPart) (PathFund v2.sol#484)
Reentrancy in PathFund.distributeLiquifiedToken(uint256) (PathFund v2.sol#470-503):
        External calls:
        - addLiquidity(tokenToLiq,liqPart / 2) (PathFund v2.sol#476)
                - uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),tokenAmount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
        - (success) = address(marketingAddress).call{value: marketingPart}() (PathFund v2.sol#482)
        - (success) = address(developmentAddress).call{value: developmentPart}() (PathFund v2.sol#490)
        Event emitted after the call(s):
        - DevelopmentCollected(developmentPart) (PathFund v2.sol#492)
Reentrancy in PathFund.distributeLiquifiedToken(uint256) (PathFund v2.sol#470-503):
        External calls:
        - addLiquidity(tokenToLiq,liqPart / 2) (PathFund v2.sol#476)
                - uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),tokenAmount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
        - (success) = address(marketingAddress).call{value: marketingPart}() (PathFund v2.sol#482)
        - (success) = address(developmentAddress).call{value: developmentPart}() (PathFund v2.sol#490)
        - (success) = address(buybackAddress).call{value: buybackPart}() (PathFund v2.sol#498)
        Event emitted after the call(s):
        - BuyBackCollected(buybackPart) (PathFund v2.sol#500)
Reentrancy in PathFund.swapAndLiquify(uint256) (PathFund v2.sol#504-508):
        External calls:
        - swapTokensForEth(amount - tokenToLiq) (PathFund v2.sol#506)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (PathFund v2.sol#443-44
9)
        - distributeLiquifiedToken(tokenToLiq) (PathFund v2.sol#507)
                - uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),tokenAmount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
                - (success) = address(marketingAddress).call{value: marketingPart}() (PathFund v2.sol#482)
                - (success) = address(developmentAddress).call{value: developmentPart}() (PathFund v2.sol#490)
                - (success) = address(buybackAddress).call{value: buybackPart}() (PathFund v2.sol#498)
        External calls sending eth:
        - distributeLiquifiedToken(tokenToLiq) (PathFund v2.sol#507)
                - uniswapV2Router.addLiquidityETH{value: bnbAmount}(address(this),tokenAmount,0,0,LPtokenReciver,block.timestamp) (PathFund v2.sol#456-463)
                - (success) = address(marketingAddress).call{value: marketingPart}() (PathFund v2.sol#482)
                - (success) = address(developmentAddress).call{value: developmentPart}() (PathFund v2.sol#490)
                - (success) = address(buybackAddress).call{value: buybackPart}() (PathFund v2.sol#498)
```



```
        - (success) = address(buybackAddress).call{value: buybackPart}() (PathFund v2.sol#498)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function IUniswapV2Router02.WETH() (PathFund v2.sol#10) is not in mixedCase
Struct PathFund.userData (PathFund v2.sol#206) is not in CapWords
Parameter PathFund.kkAirDrop(address[],uint256[])._address (PathFund v2.sol#244) is not in mixedCase
Parameter PathFund.kkAirDrop(address[],uint256[])._amount (PathFund v2.sol#244) is not in mixedCase
Parameter PathFund.transferForeignToken(address,address,uint256)._token (PathFund v2.sol#269) is not in mixedCase
Parameter PathFund.transferForeignToken(address,address,uint256)._to (PathFund v2.sol#269) is not in mixedCase
Parameter PathFund.transferForeignToken(address,address,uint256)._value (PathFund v2.sol#269) is not in mixedCase
Parameter PathFund.switchMarketActive(bool,bool)._state (PathFund v2.sol#280) is not in mixedCase
Parameter PathFund.switchMarketActive(bool,bool)._onlytransfers (PathFund v2.sol#280) is not in mixedCase
Parameter PathFund.switchBlockMultiBuys(bool,uint256)._state (PathFund v2.sol#290) is not in mixedCase
Parameter PathFund.switchLimitSells(bool)._state (PathFund v2.sol#295) is not in mixedCase
Parameter PathFund.switchLimitBuys(bool)._state (PathFund v2.sol#298) is not in mixedCase
Parameter PathFund.setFee(bool,uint256,uint256,uint256,uint256).is_buy (PathFund v2.sol#329) is not in mixedCase
Parameter PathFund.setFeeStatus(bool,bool,bool)._state (PathFund v2.sol#346) is not in mixedCase
Parameter PathFund.setSwapAndLiquify(bool,uint256,uint256,uint256)._state (PathFund v2.sol#351) is not in mixedCase
Parameter PathFund.setSwapAndLiquify(bool,uint256,uint256,uint256)._intervalSecondsForSwap (PathFund v2.sol#351) is not in mixedCase
Parameter PathFund.setSwapAndLiquify(bool,uint256,uint256,uint256)._minimumTokensBeforeSwap (PathFund v2.sol#351) is not in mixedCase
Parameter PathFund.setSwapAndLiquify(bool,uint256,uint256,uint256)._tokensToSwap (PathFund v2.sol#351) is not in mixedCase
Function PathFund.set_allFees(uint256,uint256,uint256,uint256,uint256,uint256,uint256,uint256) (PathFund v2.sol#357-361) is not in mixedCase
Function PathFund.set_maxLimits(uint256,uint256,uint256) (PathFund v2.sol#362-368) is not in mixedCase
Parameter PathFund.editPremarketUser(address,bool)._target (PathFund v2.sol#374) is not in mixedCase
Parameter PathFund.editPremarketUser(address,bool)._status (PathFund v2.sol#374) is not in mixedCase
Parameter PathFund.editExcludedFromFees(address,bool)._target (PathFund v2.sol#385) is not in mixedCase
Parameter PathFund.editExcludedFromFees(address,bool)._status (PathFund v2.sol#385) is not in mixedCase
Parameter PathFund.editAutomatedMarketMakerPairs(address,bool)._target (PathFund v2.sol#409) is not in mixedCase
Parameter PathFund.editAutomatedMarketMakerPairs(address,bool)._status (PathFund v2.sol#409) is not in mixedCase
Parameter PathFund.blacklistAccount(address,bool)._target (PathFund v2.sol#412) is not in mixedCase
Parameter PathFund.blacklistAccount(address,bool)._status (PathFund v2.sol#412) is not in mixedCase
Parameter PathFund.blacklistMultipleAccounts(address[],bool)._targets (PathFund v2.sol#424) is not in mixedCase
Parameter PathFund.blacklistMultipleAccounts(address[],bool)._state (PathFund v2.sol#424) is not in mixedCase
Function PathFund.retunList_blacklist() (PathFund v2.sol#598-602) is not in mixedCase
Function PathFund.retunList_premarket() (PathFund v2.sol#603-607) is not in mixedCase
Function PathFund.retunList_excludedFee() (PathFund v2.sol#608-612) is not in mixedCase
Function PathFund.isThisFrom_KKteam() (PathFund v2.sol#614-617) is not in mixedCase
Variable PathFund.LPtokenReciver (PathFund v2.sol#178) is not in mixedCase
Variable PathFund.total_supply (PathFund v2.sol#181) is not in mixedCase
Modifier PathFund.FastTx() (PathFund v2.sol#465-469) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (PathFund v2.sol#48)" inContext (PathFund v2.sol#43-51)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

Variable PathFund.set_allFees(uint256,uint256,uint256,uint256,uint256,uint256,uint256,uint256).bDevelopment (PathFund v2.sol#357) is too similar to PathFund
et_allFees(uint256,uint256,uint256,uint256,uint256,uint256,uint256,uint256).sDevelopment (PathFund v2.sol#357)
Variable PathFund.distributeLiquifiedToken(uint256).success_scope_0 (PathFund v2.sol#490) is too similar to PathFund.distributeLiquifiedToken(uint256).succe
_scope_1 (PathFund v2.sol#498)
Variable ERC20._totalSupply (PathFund v2.sol#83) is too similar to PathFund.total_supply (PathFund v2.sol#181)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar

PathFund.slitherConstructorVariables() (PathFund v2.sol#153-619) uses literals with too many digits:
        - maxBuyTxAmount = 300000 * 10 ** 9 (PathFund v2.sol#192)
PathFund.slitherConstructorVariables() (PathFund v2.sol#153-619) uses literals with too many digits:
        - maxSellTxAmount = 300000_0 * 10 ** 9 (PathFund v2.sol#193)
PathFund.slitherConstructorVariables() (PathFund v2.sol#153-619) uses literals with too many digits:
        - maxWallet = 3000000 * 10 ** 9 (PathFund v2.sol#194)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

PathFund.maxWalletActive (PathFund v2.sol#163) should be constant
PathFund.sellBuyBackFee (PathFund v2.sol#189) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (PathFund v2.sol#67-70)
getTime() should be declared external:
        - Ownable.getTime() (PathFund v2.sol#76-78)
name() should be declared external:
        - ERC20.name() (PathFund v2.sol#90-92)
symbol() should be declared external:
        - ERC20.symbol() (PathFund v2.sol#93-95)
totalSupply() should be declared external:
        - ERC20.totalSupply() (PathFund v2.sol#99-101)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (PathFund v2.sol#105-108)
allowance(address,address) should be declared external:
        - ERC20.allowance(address,address) (PathFund v2.sol#109-111)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (PathFund v2.sol#112-115)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (PathFund v2.sol#116-122)
setairDropAddress(address) should be declared external:
```



```
Redundant expression "this (PathFund v2.sol#48)" inContext (PathFund v2.sol#43-51)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

Variable PathFund.set_allFees(uint256,uint256,uint256,uint256,uint256,uint256,uint256,uint256).bDevelopment (PathFund v2.sol#357) is too similar to PathFund.s
et_allFees(uint256,uint256,uint256,uint256,uint256,uint256,uint256,uint256).sDevelopment (PathFund v2.sol#357)
Variable PathFund.distributeLiquifiedToken(uint256).success_scope_0 (PathFund v2.sol#490) is too similar to PathFund.distributeLiquifiedToken(uint256).success
_scope_1 (PathFund v2.sol#498)
Variable ERC20._totalSupply (PathFund v2.sol#83) is too similar to PathFund.total_supply (PathFund v2.sol#181)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar

PathFund.slitherConstructorVariables() (PathFund v2.sol#153-619) uses literals with too many digits:
        - maxBuyTxAmount = 300000 * 10 ** 9 (PathFund v2.sol#192)
PathFund.slitherConstructorVariables() (PathFund v2.sol#153-619) uses literals with too many digits:
        - maxSellTxAmount = 300000_0 * 10 ** 9 (PathFund v2.sol#193)
PathFund.slitherConstructorVariables() (PathFund v2.sol#153-619) uses literals with too many digits:
        - maxWallet = 3000000 * 10 ** 9 (PathFund v2.sol#194)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

PathFund.maxWalletActive (PathFund v2.sol#163) should be constant
PathFund.sellBuyBackFee (PathFund v2.sol#189) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (PathFund v2.sol#67-70)
getTime() should be declared external:
        - Ownable.getTime() (PathFund v2.sol#76-78)
name() should be declared external:
        - ERC20.name() (PathFund v2.sol#90-92)
symbol() should be declared external:
        - ERC20.symbol() (PathFund v2.sol#93-95)
totalSupply() should be declared external:
        - ERC20.totalSupply() (PathFund v2.sol#99-101)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (PathFund v2.sol#105-108)
allowance(address,address) should be declared external:
        - ERC20.allowance(address,address) (PathFund v2.sol#109-111)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (PathFund v2.sol#112-115)
transferFrom(address,address,uint256) should be declared external:
        - ERC20.transferFrom(address,address,uint256) (PathFund v2.sol#116-122)
setairDropAddress(address) should be declared external:
        - PathFund.setairDropAddress(address) (PathFund v2.sol#253-255)
setFeesAddress(address,address,address,address) should be declared external:
        - PathFund.setFeesAddress(address,address,address,address) (PathFund v2.sol#309-317)
betterTransferOwnership(address) should be declared external:
        - PathFund.betterTransferOwnership(address) (PathFund v2.sol#319-327)
setSwapAndLiquify(bool,uint256,uint256,uint256) should be declared external:
        - PathFund.setSwapAndLiquify(bool,uint256,uint256,uint256) (PathFund v2.sol#351-356)
set_allFees(uint256,uint256,uint256,uint256,uint256,uint256,uint256,uint256) should be declared external:
        - PathFund.set_allFees(uint256,uint256,uint256,uint256,uint256,uint256,uint256,uint256) (PathFund v2.sol#357-361)
set_maxLimits(uint256,uint256,uint256) should be declared external:
        - PathFund.set_maxLimits(uint256,uint256,uint256) (PathFund v2.sol#362-368)
excludeMultipleAccountsFromFees(address[],bool) should be declared external:
        - PathFund.excludeMultipleAccountsFromFees(address[],bool) (PathFund v2.sol#396-408)
retunList_blacklist() should be declared external:
        - PathFund.retunList_blacklist() (PathFund v2.sol#598-602)
retunList_premarket() should be declared external:
        - PathFund.retunList_premarket() (PathFund v2.sol#603-607)
retunList_excludedFee() should be declared external:
        - PathFund.retunList_excludedFee() (PathFund v2.sol#608-612)
isThisFrom_KKteam() should be declared external:
        - PathFund.isThisFrom_KKteam() (PathFund v2.sol#614-617)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
PathFund v2.sol analyzed (8 contracts with 75 detectors), 109 result(s) found
```

<u>Results:</u> Major issues are not found.

Closing Summary

Overall, smart contracts are very well written and adhere to guidelines.Good to deploy on BSC chain.

Disclaimer

Quillhash audit is not a security warranty, investment advice, or endorsement of the **PathFund platform**. This audit does not provide a security or correctness guarantee of the audited smart contracts. The statements made in this document should not be interpreted as investment or legal advice, nor should its authors be held accountable for decisions made based on them. Securing smart contracts is a multistep process. One audit cannot be considered enough. We recommend that the PathFund Team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.