QuillAudits

Audit Report
September, 2021

HYBRID DOGE

Contents

Scope of Audit

The scope of this audit was to analyze and document the Hybrid Doge CoinToken smart contract codebase for quality, security, and correctness.

Checked Vulnerabilities

We have scanned the smart contract for commonly known and more specific vulnerabilities. Here are some of the commonly known vulnerabilities that we considered:

- 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 math
- Unsafe type inference
- Implicit visibility level

Techniques and Methods

Throughout the audit of smart contract, 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.
SmartCheck.

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

Code Review / Manual Analysis
Manual Analysis or review of code was done to identify new vulnerability 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 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 possibilities of optimization of code to reduce gas consumption.

Tools and Platforms used for Audit
Remix IDE, Truffle, Truffle Team, Ganache, Solhint, Mythril, Slither, SmartCheck.

Issue Categories

Every issue in this report has been assigned with 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 to be fixed before moving to a live environment.

Medium level 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 level 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

These are severity four issues which 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 issues per severity

Type	High	Medium	Low	Informational
Open	0	0	5	7
Acknowledged	0	0	0	0
Closed	0	0	0	0

Introduction

During the period of **September 16, 2021, to September 17, 2021** - QuillAudits Team performed a security audit for the Hybrid Doge V2 smart contract.

The code for the audit was taken from following the official link: https://bscscan.com/address/0x9E8bD931610b71Fc4e2190D6102fb510e4995546#code

Issues Found – Code Review / Manual Testing

High severity issues

No issues were found.

Medium severity issues

No issues were found.

Low level severity issues

1. **Possible to gain ownership**

   Description
   Possible to gain ownership after renouncing the contract ownership. The owner can renounce ownership and make a contract without an owner, but he can regain ownership by following the steps below:
   1. Owner calls the lock function in contract to set the current owner as _previousOwner.
   2. Owner calls unlock to unlock the contract and set _owner = _previousOwner.
   3. Owner called renounceOwnership to leave the contract without the owner.
   4. Owner calls unlock to regain ownership.

   Remediation
   We suggest removing these lock/unlock functions as this seems not to serve a great purpose. Otherwise, always renounce ownership first before calling the lock function.

   Status: Open

2. Infinite loop

Line	Code

612	function includeInReward(address account) external onlyOwner() {
	require(_isExcluded[account], "Account is already included");
	for (uint256 i = 0; i < _excluded.length; i++) {
	if (_excluded[i] == account) {
	_excluded[i] = _excluded[_excluded.length - 1];
	_tOwned[account] = 0;
	_isExcluded[account] = false;
	_excluded.pop();
	break;
	}
	}
	}

Description

In includeInReward & _getCurrentSupply functions for loop do not have _excluded length limit , which costs more gas.

Remediation

_exclude length should be limited.

Status: Open

3. Function input parameters lack of check

Line	Code

644	644 function setTaxFeePercent(uint256 taxFee) external onlyOwner() {
	645 _taxFee = taxFee;
	646 }
	647
	648 function setDevFeePercent(uint256 devFee) external onlyOwner() {
	649 _devFee = devFee;
	650 }
	651
	652 function setLiquidityFeePercent(uint256 liquidityFee) external onlyOwner() {
	653 _liquidityFee = liquidityFee;
	654 }

Description

Check the values passed in the function is not < 0 and its not > 100 because the Percentage is between 0 to 100 only.

Remediation
Line no : 644, 648, 652 following functions need to check for input params must be between 0 and 100, setTaxFeePercent, setDevFeePercent, setLiquidityFeePercent.

Status: Open

4. Centralized risk in addLiquidity

Line	Code

841	function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private
	{
	_approve(address(this), address(uniswapV2Router),tokenAmount);
	uniswapV2Router.addLiquidityETH{value: ethAmount}(
	address(this),
	tokenAmount,
	0, // slippage is unavoidable
	0, // slippage is unavoidable
	owner(),
	block.timestamp
	);
	}

Description
In the addLiquidityETH function, the owner gets Tokens from the Pool. If the private key of the owner's wallet is compromised, then it will create a problem.

Remediation
Ideally, this can be a governance smart contract. On another hand, the owner can accept this risk and handle the private key very securely.

Status: Open

5. Centralization Risks

Description
Detects missing zero address validation. Constructor has routerAddress, fee address, tokenOwner, and service, check if this address is valid or not. There are many functions needed to add zero address checks.

Remediation
Check that the address is not zero. excludeFromReward, includeInReward, excludeFromFee, includeInFee, setDevWalletAddress, setRouterAddress.

Status: Open

Informational

6. Critical operation lacks event log

Description
Missing event log for: deliver, setTaxFeePercent, setDevFeePercent, setLiquidityFeePercent, setMaxTxPercent , setDevWalletAddress addLiquidity,removeAllFee, restoreAllFee.

Remediation
Please write an event log for listed events

Status: Open

7. Spelling mistakes

Line	Code
670	//to recieve ETH from uniswapV2Router when swaping receive() external payable {}

Description
Typing mistakes in comments.

Remediation
Please correct the spelling.

Status: Open

8. **Presence of unused code**

Description

If any function is not called from inside the smart contract, then it is better to declare it as external instead of public. As it saves some gas as well.
https://ethereum.stackexchange.com/questions/19380/external-vs-public-best-practices

Status: Open

9. **Ambiguous function and the parameter name**

Line	Code

656	function setMaxTxPercent(uint256 maxTxPercent) public onlyOwner {
	_maxTxAmount = maxTxPercent * 10 ** _decimals;
	}

Description

This function name and parameter indicates to set percentage for maxTxPercent, but it sets _maxTxAmount for the contract.

Remediation

Variable and function names should be relevant to the task done inside the function.

Status: Open

10. **State variables that could be declared constant**

Line	Code
451-452	string private _name; string private _symbol;

Description

Constant state variables should be declared constant to save gas.

Remediation

Line no 451,452 Add the constant attributes to state variables that never change.

Status: Open

11. State variables that could be declared constant

Line	Code
480	constructor (string memory _NAME, string memory _SYMBOL, uint256 _DECIMALS, uint256 _supply, uint256 _txFee,uint256 _lpFee,uint256 _DexFee,address routerAddress,address feeaddress,address tokenOwner,address service) public payable { _name = _NAME;

Description
Visibility of the constructor is ignored.

Remediation
Line no: 480 remove word public from constructor.

Status: Open

12. Unused event

Line	Code
480	event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);

Description
This event has not been used in code.

Remediation
Remove unused code.

Status: Open

Functional test

Function Names	Testing results
constructor	Visibility of constructor ignored
name	Passed
symbol	Passed
decimals	Passed
totalSupply	Passed
balanceOf	Passed
transfer	Passed
allowance	Passed
approve	Passed
transferFrom	Passed
increaseAllowance	Passed
decreaseAllowance	Passed
isExcludedFromReward	Passed
totalFees	Passed
deliver	Function input parameters lack of check
reflectionFromToken	Passed
tokenFromReflection	Passed
excludeFromReward	Missing zero address validation
includeInReward	Infinite loop
_transferBothExcluded	Passed

Function Names	Testing results
excludeFromFee	Missing zero address validation
includeInFee	Missing zero address validation
setTaxFeePercent	Function input parameters lack of check
setDevFeePercent	Function input parameters lack of check
setLiquidityFeePercent	Function input parameters lack of check
setMaxTxPercent	Critical operation lacks event log
setDevWalletAddress	Critical operation lacks event log
setSwapAndLiquifyEnabled	Passed
receive	Passed
_reflectFee	Passed
_getValues	Passed
_getTValues	Passed
_getRValues	Passed
_getRate	Passed
_getCurrentSupply	Infinite loop
_takeLiquidity	Passed
_takeDev	Passed
calculateTaxFee	Passed
calculateDevFee	Passed
calculateLiquidityFee	Passed
removeAllFee	Critical operation lacks event log

Function Names	Testing results
restoreAllFee	Critical operation lacks event log
isExcludedFromFee	Passed
_approve	Passed
_transfer	Passed
swapAndLiquify	Passed
swapTokensForEth	Passed
addLiquidity	Centralized risk in addLiquidity
_tokenTransfer	Passed
_transferStandard	Passed
_transferToExcluded	Passed
_transferFromExcluded	Passed
setRouterAddress	Function input parameters lack of check
setNumTokensSellToAddToLiquidity	Passed
owner	Passed
onlyOwner	Passed
renounceOwnership	Possible to gain ownership
transferOwnership	Passed
lock	Passed
unlock	Passed
_msgSender	Passed
_msgData	Passed

Automated Testing

Slither

```
INFO:Detectors:
CoinToken.addLiquidity(uint256,uint256) (CoinToken.sol#841-851) sends eth to arbitrary user
        Dangerous calls:
        - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol#843-850
)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#functions-that-send-ether-to-arbitrary-destinations
INFO:Detectors:
Reentrancy in CoinToken._transfer(address,address,uint256) (CoinToken.sol#780-815):
        External calls:
        - swapAndLiquify(contractTokenBalance) (CoinToken.sol#806)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#843-850)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (C
oinToken.sol#832-838)
        External calls sending eth:
        - swapAndLiquify(contractTokenBalance) (CoinToken.sol#806)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#843-850)
        State variables written after the call(s):
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#814)
                - _rOwned[_devWalletAddress] = _rOwned[_devWalletAddress].add(rDev) (CoinToken.sol#729)
                - _rOwned[address(this)] = _rOwned[address(this)].add(rLiquidity) (CoinToken.sol#721)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (CoinToken.sol#885)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (CoinToken.sol#875)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (CoinToken.sol#627)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (CoinToken.sol#876)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (CoinToken.sol#897)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (CoinToken.sol#887)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (CoinToken.sol#898)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (CoinToken.sol#629)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#814)
                - _rTotal = _rTotal.sub(rFee) (CoinToken.sol#674)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#814)
                - _tOwned[_devWalletAddress] = _tOwned[_devWalletAddress].add(tDev) (CoinToken.sol#731)
                - _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (CoinToken.sol#723)
                - _tOwned[sender] = _tOwned[sender].sub(tAmount) (CoinToken.sol#896)
                - _tOwned[sender] = _tOwned[sender].sub(tAmount) (CoinToken.sol#626)
```

```
                - _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (CoinToken.sol#723)
                - _tOwned[sender] = _tOwned[sender].sub(tAmount) (CoinToken.sol#896)
                - _tOwned[sender] = _tOwned[sender].sub(tAmount) (CoinToken.sol#626)
                - _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (CoinToken.sol#886)
                - _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (CoinToken.sol#628)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities
INFO:Detectors:
CoinToken.constructor(string,string,uint256,uint256,uint256,uint256,uint256,address,address,address,address) (CoinToken.sol#480-516) perf
orms a multiplication on the result of a division:
        -_maxTxAmount = (_tTotal * 5 / 1000) * 10 ** _decimals (CoinToken.sol#493)
CoinToken.constructor(string,string,uint256,uint256,uint256,uint256,address,address,address,address) (CoinToken.sol#480-516) perf
orms a multiplication on the result of a division:
        -numTokensSellToAddToLiquidity = (_tTotal * 5 / 10000) * 10 ** _decimals (CoinToken.sol#494)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply
INFO:Detectors:
CoinToken.addLiquidity(uint256,uint256) (CoinToken.sol#841-851) ignores return value by uniswapV2Router.addLiquidityETH{value: ethAmount}
(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol#843-850)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
CoinToken.allowance(address,address).owner (CoinToken.sol#544) shadows:
        - Ownable.owner() (CoinToken.sol#208-210) (function)
CoinToken._approve(address,address,uint256).owner (CoinToken.sol#772) shadows:
        - Ownable.owner() (CoinToken.sol#208-210) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
CoinToken.constructor(string,string,uint256,uint256,uint256,uint256,uint256,address,address,address,address).feeaddress (CoinToken.sol#48
0) lacks a zero-check on :
                - _devWalletAddress = feeaddress (CoinToken.sol#495)
CoinToken.constructor(string,string,uint256,uint256,uint256,uint256,uint256,address,address,address,address).tokenOwner (CoinToken.sol#48
0) lacks a zero-check on :
                - _owner = tokenOwner (CoinToken.sol#511)
CoinToken.constructor(string,string,uint256,uint256,uint256,uint256,uint256,address,address,address,address).service (CoinToken.sol#480)
lacks a zero-check on :
                - address(service).transfer(msg.value) (CoinToken.sol#512)
CoinToken.setDevWalletAddress(address)._addr (CoinToken.sol#660) lacks a zero-check on :
                - _devWalletAddress = _addr (CoinToken.sol#661)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
```

```
INFO:Detectors:
Reentrancy in CoinToken._transfer(address,address,uint256) (CoinToken.sol#780-815):
        External calls:
        - swapAndLiquify(contractTokenBalance) (CoinToken.sol#806)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#843-850)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (C
oinToken.sol#832-838)
        External calls sending eth:
        - swapAndLiquify(contractTokenBalance) (CoinToken.sol#806)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#843-850)
        State variables written after the call(s):
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#814)
                - _devFee = _previousDevFee (CoinToken.sol#764)
                - _devFee = 0 (CoinToken.sol#758)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#814)
                - _liquidityFee = _previousLiquidityFee (CoinToken.sol#765)
                - _liquidityFee = 0 (CoinToken.sol#759)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#814)
                - _previousDevFee = _devFee (CoinToken.sol#754)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#814)
                - _previousLiquidityFee = _liquidityFee (CoinToken.sol#755)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#814)
                - _previousTaxFee = _taxFee (CoinToken.sol#753)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#814)
                - _tFeeTotal = _tFeeTotal.add(tFee) (CoinToken.sol#675)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#814)
                - _taxFee = _previousTaxFee (CoinToken.sol#763)
                - _taxFee = 0 (CoinToken.sol#757)
Reentrancy in CoinToken.constructor(string,string,uint256,uint256,uint256,uint256,uint256,address,address,address,address) (CoinToken.sol
#480-516):
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (CoinToken.sol#
501-502)
        State variables written after the call(s):
        - _isExcludedFromFee[tokenOwner] = true (CoinToken.sol#508)
        - _isExcludedFromFee[address(this)] = true (CoinToken.sol#509)
```

```
        - _isExcludedFromFee[tokenOwner] = true (CoinToken.sol#508)
        - _isExcludedFromFee[address(this)] = true (CoinToken.sol#509)
        - _owner = tokenOwner (CoinToken.sol#511)
        - uniswapV2Router = _uniswapV2Router (CoinToken.sol#505)
Reentrancy in CoinToken.setRouterAddress(address) (CoinToken.sol#906-910):
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (CoinToken.sol#
908)
        State variables written after the call(s):
        - uniswapV2Router = _uniswapV2Router (CoinToken.sol#909)
Reentrancy in CoinToken.swapAndLiquify(uint256) (CoinToken.sol#817-825):
        External calls:
        - swapTokensForEth(half) (CoinToken.sol#821)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (C
oinToken.sol#832-838)
        - addLiquidity(otherHalf,newBalance) (CoinToken.sol#823)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#843-850)
        External calls sending eth:
        - addLiquidity(otherHalf,newBalance) (CoinToken.sol#823)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#843-850)
        State variables written after the call(s):
        - addLiquidity(otherHalf,newBalance) (CoinToken.sol#823)
                - _allowances[owner][spender] = amount (CoinToken.sol#776)
Reentrancy in CoinToken.transferFrom(address,address,uint256) (CoinToken.sol#553-557):
        External calls:
        - _transfer(sender,recipient,amount) (CoinToken.sol#554)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#843-850)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (C
oinToken.sol#832-838)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (CoinToken.sol#554)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#843-850)
        State variables written after the call(s):
        - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (CoinToken
```

```
        - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (CoinToken
.sol#555)
                - _allowances[owner][spender] = amount (CoinToken.sol#776)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in CoinToken._transfer(address,address,uint256) (CoinToken.sol#780-815):
        External calls:
        - swapAndLiquify(contractTokenBalance) (CoinToken.sol#806)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#843-850)
```

```
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (C
oinToken.sol#832-838)
        External calls sending eth:
        - swapAndLiquify(contractTokenBalance) (CoinToken.sol#806)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#843-850)
        Event emitted after the call(s):
        - Transfer(sender,recipient,tTransferAmount) (CoinToken.sol#880)
                - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#814)
        - Transfer(sender,recipient,tTransferAmount) (CoinToken.sol#902)
                - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#814)
        - Transfer(sender,recipient,tTransferAmount) (CoinToken.sol#891)
                - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#814)
        - Transfer(sender,recipient,tTransferAmount) (CoinToken.sol#633)
                - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#814)
Reentrancy in CoinToken.constructor(string,string,uint256,uint256,uint256,uint256,uint256,address,address,address,address) (CoinToken.sol
#480-516):
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (CoinToken.sol#
501-502)
        External calls sending eth:
        - address(service).transfer(msg.value) (CoinToken.sol#512)
        Event emitted after the call(s):
        - Transfer(address(0),tokenOwner,_tTotal) (CoinToken.sol#513)
Reentrancy in CoinToken.swapAndLiquify(uint256) (CoinToken.sol#817-825):
        External calls:
        - swapTokensForEth(half) (CoinToken.sol#821)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (C
oinToken.sol#832-838)
        - addLiquidity(otherHalf,newBalance) (CoinToken.sol#823)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#843-850)
        External calls sending eth:
        - addLiquidity(otherHalf,newBalance) (CoinToken.sol#823)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#843-850)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (CoinToken.sol#777)
```

```
        - Approval(owner,spender,amount) (CoinToken.sol#777)
                - addLiquidity(otherHalf,newBalance) (CoinToken.sol#823)
        - SwapAndLiquify(half,newBalance,otherHalf) (CoinToken.sol#824)
Reentrancy in CoinToken.transferFrom(address,address,uint256) (CoinToken.sol#553-557):
        External calls:
        - _transfer(sender,recipient,amount) (CoinToken.sol#554)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#843-850)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (C
oinToken.sol#832-838)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (CoinToken.sol#554)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#843-850)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (CoinToken.sol#777)
                - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (C
oinToken.sol#555)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
Ownable.unlock() (CoinToken.sol#239-244) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp > _lockTime,Contract is locked.) (CoinToken.sol#241)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Address.isContract(address) (CoinToken.sol#127-131) uses assembly
        - INLINE ASM (CoinToken.sol#129)
Address._verifyCallResult(bool,bytes,string) (CoinToken.sol#179-192) uses assembly
        - INLINE ASM (CoinToken.sol#184-187)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address._verifyCallResult(bool,bytes,string) (CoinToken.sol#179-192) is never used and should be removed
Address.functionCall(address,bytes) (CoinToken.sol#139-141) is never used and should be removed
Address.functionCall(address,bytes,string) (CoinToken.sol#143-145) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (CoinToken.sol#147-149) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (CoinToken.sol#151-156) is never used and should be removed
Address.functionDelegateCall(address,bytes) (CoinToken.sol#169-171) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (CoinToken.sol#173-177) is never used and should be removed
```

```
Address.functionStaticCall(address,bytes) (CoinToken.sol#158-160) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (CoinToken.sol#162-166) is never used and should be removed
Address.isContract(address) (CoinToken.sol#127-131) is never used and should be removed
Address.sendValue(address,uint256) (CoinToken.sol#133-137) is never used and should be removed
Context._msgData() (CoinToken.sol#118-121) is never used and should be removed
SafeMath.div(uint256,uint256,string) (CoinToken.sol#95-100) is never used and should be removed
SafeMath.mod(uint256,uint256) (CoinToken.sol#84-86) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (CoinToken.sol#102-107) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (CoinToken.sol#24-30) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (CoinToken.sol#51-56) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (CoinToken.sol#58-63) is never used and should be removed
```

```
SafeMath.tryMod(uint256,uint256) (CoinToken.sol#58-63) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (CoinToken.sol#39-49) is never used and should be removed
SafeMath.trySub(uint256,uint256) (CoinToken.sol#32-37) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.4 (CoinToken.sol#7) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.4 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (CoinToken.sol#133-137):
        - (success) = recipient.call{value: amount}() (CoinToken.sol#135)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (CoinToken.sol#151-156):
        - (success,returndata) = target.call{value: value}(data) (CoinToken.sol#154)
Low level call in Address.functionStaticCall(address,bytes,string) (CoinToken.sol#162-166):
        - (success,returndata) = target.staticcall(data) (CoinToken.sol#164)
Low level call in Address.functionDelegateCall(address,bytes,string) (CoinToken.sol#173-177):
        - (success,returndata) = target.delegatecall(data) (CoinToken.sol#175)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Variable Ownable._owner (CoinToken.sol#198) is not in mixedCase
Variable Ownable._lockTime (CoinToken.sol#200) is not in mixedCase
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (CoinToken.sol#271) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (CoinToken.sol#272) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (CoinToken.sol#286) is not in mixedCase
Function IUniswapV2Router01.WETH() (CoinToken.sol#304) is not in mixedCase
Parameter CoinToken.setDevWalletAddress(address)._addr (CoinToken.sol#660) is not in mixedCase
Parameter CoinToken.setSwapAndLiquifyEnabled(bool)._enabled (CoinToken.sol#665) is not in mixedCase
Parameter CoinToken.calculateTaxFee(uint256)._amount (CoinToken.sol#734) is not in mixedCase
Parameter CoinToken.calculateDevFee(uint256)._amount (CoinToken.sol#740) is not in mixedCase
Parameter CoinToken.calculateLiquidityFee(uint256)._amount (CoinToken.sol#746) is not in mixedCase
Variable CoinToken._devWalletAddress (CoinToken.sol#446) is not in mixedCase
Variable CoinToken._taxFee (CoinToken.sol#454) is not in mixedCase
Variable CoinToken._devFee (CoinToken.sol#456) is not in mixedCase
Variable CoinToken._liquidityFee (CoinToken.sol#458) is not in mixedCase
Variable CoinToken._maxTxAmount (CoinToken.sol#464) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (CoinToken.sol#119)" inContext (CoinToken.sol#113-122)
```

```
INFO:Detectors:
Redundant expression "this (CoinToken.sol#119)" inContext (CoinToken.sol#113-122)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
Reentrancy in CoinToken.constructor(string,string,uint256,uint256,uint256,uint256,uint256,address,address,address,address) (CoinToken.sol
#480-516):
        External calls:
        - address(service).transfer(msg.value) (CoinToken.sol#512)
        Event emitted after the call(s):
        - Transfer(address(0),tokenOwner,_tTotal) (CoinToken.sol#513)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (CoinToken.sol#3
08) is too similar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (Co
inToken.sol#309)
Variable CoinToken._getValues(uint256).rTransferAmount (CoinToken.sol#680) is too similar to CoinToken._transferToExcluded(address,addres
s,uint256).tTransferAmount (CoinToken.sol#884)
Variable CoinToken._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (CoinToken.sol#697) is too similar to CoinToken._
transferToExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#884)
Variable CoinToken._transferBothExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#625) is too similar to CoinToken._transf
erBothExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#625)
Variable CoinToken._transferBothExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#625) is too similar to CoinToken._getTVa
lues(uint256).tTransferAmount (CoinToken.sol#688)
Variable CoinToken._transferBothExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#625) is too similar to CoinToken._getVal
ues(uint256).tTransferAmount (CoinToken.sol#679)
Variable CoinToken._getValues(uint256).rTransferAmount (CoinToken.sol#680) is too similar to CoinToken._transferFromExcluded(address,addr
ess,uint256).tTransferAmount (CoinToken.sol#895)
Variable CoinToken._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (CoinToken.sol#697) is too similar to CoinToken._
transferFromExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#895)
Variable CoinToken._transferBothExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#625) is too similar to CoinToken._transf
erStandard(address,address,uint256).tTransferAmount (CoinToken.sol#874)
Variable CoinToken.reflectionFromToken(uint256,bool).rTransferAmount (CoinToken.sol#592) is too similar to CoinToken._transferToExcluded(
address,address,uint256).tTransferAmount (CoinToken.sol#884)
Variable CoinToken._transferFromExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#895) is too similar to CoinToken._transf
erFromExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#895)
Variable CoinToken._transferBothExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#625) is too similar to CoinToken._transf
erToExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#884)
Variable CoinToken._transferToExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#884) is too similar to CoinToken._transfer
```

```
Variable CoinToken._transferToExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#884) is too similar to CoinToken._transfer
ToExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#884)
Variable CoinToken._getValues(uint256).rTransferAmount (CoinToken.sol#680) is too similar to CoinToken._getTValues(uint256).tTransferAmou
nt (CoinToken.sol#688)
Variable CoinToken._transferBothExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#625) is too similar to CoinToken._transf
erFromExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#895)
Variable CoinToken.reflectionFromToken(uint256,bool).rTransferAmount (CoinToken.sol#592) is too similar to CoinToken._getTValues(uint256)
.tTransferAmount (CoinToken.sol#688)
Variable CoinToken.reflectionFromToken(uint256,bool).rTransferAmount (CoinToken.sol#592) is too similar to CoinToken._transferFromExclude
d(address,address,uint256).tTransferAmount (CoinToken.sol#895)
```

```
Variable CoinToken.reflectionFromToken(uint256,bool).rTransferAmount (CoinToken.sol#592) is too similar to CoinToken._getTValues(uint256)
.tTransferAmount (CoinToken.sol#688)
Variable CoinToken.reflectionFromToken(uint256,bool).rTransferAmount (CoinToken.sol#592) is too similar to CoinToken._transferFromExclude
d(address,address,uint256).tTransferAmount (CoinToken.sol#895)
Variable CoinToken._transferFromExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#895) is too similar to CoinToken._transf
erToExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#884)
Variable CoinToken._transferFromExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#895) is too similar to CoinToken._getTVa
lues(uint256).tTransferAmount (CoinToken.sol#688)
Variable CoinToken._transferFromExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#895) is too similar to CoinToken._transf
erStandard(address,address,uint256).tTransferAmount (CoinToken.sol#874)
Variable CoinToken.reflectionFromToken(uint256,bool).rTransferAmount (CoinToken.sol#592) is too similar to CoinToken._transferBothExclude
d(address,address,uint256).tTransferAmount (CoinToken.sol#625)
Variable CoinToken._transferStandard(address,address,uint256).rTransferAmount (CoinToken.sol#874) is too similar to CoinToken._getTValues
(uint256).tTransferAmount (CoinToken.sol#688)
Variable CoinToken._transferFromExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#895) is too similar to CoinToken._getVal
ues(uint256).tTransferAmount (CoinToken.sol#679)
Variable CoinToken._getValues(uint256).rTransferAmount (CoinToken.sol#680) is too similar to CoinToken._transferStandard(address,address,
uint256).tTransferAmount (CoinToken.sol#874)
Variable CoinToken._getRValues(uint256,uint256,uint256,uint256,uint256).rTransferAmount (CoinToken.sol#697) is too similar to CoinToken._
transferStandard(address,address,uint256).tTransferAmount (CoinToken.sol#874)
Variable CoinToken._transferStandard(address,address,uint256).rTransferAmount (CoinToken.sol#874) is too similar to CoinToken._transferBo
thExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#625)
Variable CoinToken._transferStandard(address,address,uint256).rTransferAmount (CoinToken.sol#874) is too similar to CoinToken._getValues(
uint256).tTransferAmount (CoinToken.sol#679)
Variable CoinToken.reflectionFromToken(uint256,bool).rTransferAmount (CoinToken.sol#592) is too similar to CoinToken._transferStandard(ad
dress,address,uint256).tTransferAmount (CoinToken.sol#874)
Variable CoinToken._transferFromExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#895) is too similar to CoinToken._transf
erBothExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#625)
Variable CoinToken._transferToExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#884) is too similar to CoinToken._transfer
FromExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#895)
Variable CoinToken.reflectionFromToken(uint256,bool).rTransferAmount (CoinToken.sol#592) is too similar to CoinToken._getValues(uint256).
tTransferAmount (CoinToken.sol#679)
Variable CoinToken._transferStandard(address,address,uint256).rTransferAmount (CoinToken.sol#874) is too similar to CoinToken._transferFr
omExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#895)
Variable CoinToken._transferStandard(address,address,uint256).rTransferAmount (CoinToken.sol#874) is too similar to CoinToken._transferSt
andard(address,address,uint256).tTransferAmount (CoinToken.sol#874)
Variable CoinToken._getRValues(uint256,uint256,uint256,uint256,uint256).rTransferAmount (CoinToken.sol#697) is too similar to CoinToken._
```

```
Variable CoinToken._getRValues(uint256,uint256,uint256,uint256,uint256).rTransferAmount (CoinToken.sol#697) is too similar to CoinToken._
transferBothExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#625)
Variable CoinToken._getValues(uint256).rTransferAmount (CoinToken.sol#680) is too similar to CoinToken._getValues(uint256).tTransferAmoun
t (CoinToken.sol#679)
Variable CoinToken._getRValues(uint256,uint256,uint256,uint256,uint256).rTransferAmount (CoinToken.sol#697) is too similar to CoinToken._
getValues(uint256).tTransferAmount (CoinToken.sol#679)
Variable CoinToken._transferToExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#884) is too similar to CoinToken._getValue
s(uint256).tTransferAmount (CoinToken.sol#679)
Variable CoinToken._transferToExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#884) is too similar to CoinToken._transfer
BothExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#625)
Variable CoinToken._transferStandard(address,address,uint256).rTransferAmount (CoinToken.sol#874) is too similar to CoinToken._transferTo
Excluded(address,address,uint256).tTransferAmount (CoinToken.sol#884)
Variable CoinToken._getRValues(uint256,uint256,uint256,uint256,uint256).rTransferAmount (CoinToken.sol#697) is too similar to CoinToken._
getTValues(uint256).tTransferAmount (CoinToken.sol#688)
Variable CoinToken._transferToExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#884) is too similar to CoinToken._getTValu
es(uint256).tTransferAmount (CoinToken.sol#688)
Variable CoinToken._getValues(uint256).rTransferAmount (CoinToken.sol#680) is too similar to CoinToken._transferBothExcluded(address,addr
ess,uint256).tTransferAmount (CoinToken.sol#625)
Variable CoinToken._transferToExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#884) is too similar to CoinToken._transfer
Standard(address,address,uint256).tTransferAmount (CoinToken.sol#874)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (CoinToken.sol#217-220)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (CoinToken.sol#223-227)
lock(uint256) should be declared external:
        - Ownable.lock(uint256) (CoinToken.sol#231-236)
unlock() should be declared external:
        - Ownable.unlock() (CoinToken.sol#239-244)
name() should be declared external:
        - CoinToken.name() (CoinToken.sol#518-520)
symbol() should be declared external:
        - CoinToken.symbol() (CoinToken.sol#522-524)
decimals() should be declared external:
        - CoinToken.decimals() (CoinToken.sol#526-528)
totalSupply() should be declared external:
```

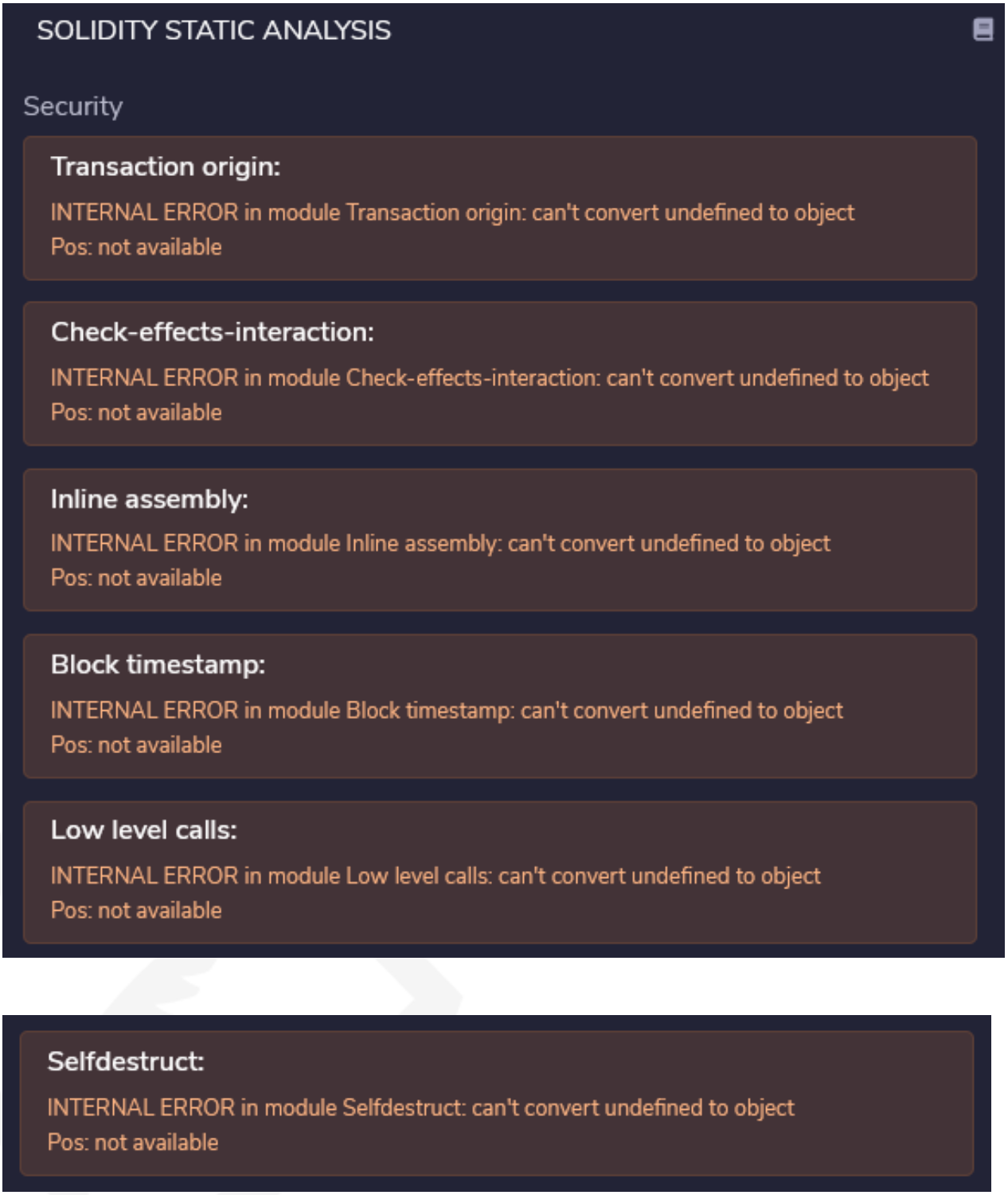
```
totalSupply() should be declared external:
        - CoinToken.totalSupply() (CoinToken.sol#530-532)
transfer(address,uint256) should be declared external:
        - CoinToken.transfer(address,uint256) (CoinToken.sol#539-542)
allowance(address,address) should be declared external:
        - CoinToken.allowance(address,address) (CoinToken.sol#544-546)
approve(address,uint256) should be declared external:
        - CoinToken.approve(address,uint256) (CoinToken.sol#548-551)
transferFrom(address,address,uint256) should be declared external:
        - CoinToken.transferFrom(address,address,uint256) (CoinToken.sol#553-557)
increaseAllowance(address,uint256) should be declared external:
        - CoinToken.increaseAllowance(address,uint256) (CoinToken.sol#559-562)
```

```
decreaseAllowance(address,uint256) should be declared external:
        - CoinToken.decreaseAllowance(address,uint256) (CoinToken.sol#564-567)
isExcludedFromReward(address) should be declared external:
        - CoinToken.isExcludedFromReward(address) (CoinToken.sol#569-571)
totalFees() should be declared external:
        - CoinToken.totalFees() (CoinToken.sol#573-575)
deliver(uint256) should be declared external:
        - CoinToken.deliver(uint256) (CoinToken.sol#577-584)
reflectionFromToken(uint256,bool) should be declared external:
        - CoinToken.reflectionFromToken(uint256,bool) (CoinToken.sol#586-595)
excludeFromReward(address) should be declared external:
        - CoinToken.excludeFromReward(address) (CoinToken.sol#603-610)
excludeFromFee(address) should be declared external:
        - CoinToken.excludeFromFee(address) (CoinToken.sol#636-638)
includeInFee(address) should be declared external:
        - CoinToken.includeInFee(address) (CoinToken.sol#640-642)
setMaxTxPercent(uint256) should be declared external:
        - CoinToken.setMaxTxPercent(uint256) (CoinToken.sol#656-658)
setDevWalletAddress(address) should be declared external:
        - CoinToken.setDevWalletAddress(address) (CoinToken.sol#660-662)
setSwapAndLiquifyEnabled(bool) should be declared external:
        - CoinToken.setSwapAndLiquifyEnabled(bool) (CoinToken.sol#665-668)
isExcludedFromFee(address) should be declared external:
        - CoinToken.isExcludedFromFee(address) (CoinToken.sol#768-770)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:CoinToken.sol analyzed (10 contracts with 75 detectors), 135 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Results

No major issues were found. Some false positive errors were reported by the tool. All the other issues have been categorized above according to their level of severity.

SOLIDITY STATIC ANALYSIS



Gas & Economy

This on local calls:

INTERNAL ERROR in module This on local calls: can't convert undefined to object
Pos: not available

Delete dynamic array:

INTERNAL ERROR in module Delete dynamic array: can't convert undefined to object
Pos: not available

For loop over dynamic array:

INTERNAL ERROR in module For loop over dynamic array: can't convert undefined to object
Pos: not available

Ether transfer in loop:

INTERNAL ERROR in module Ether transfer in loop: can't convert undefined to object
Pos: not available

ERC

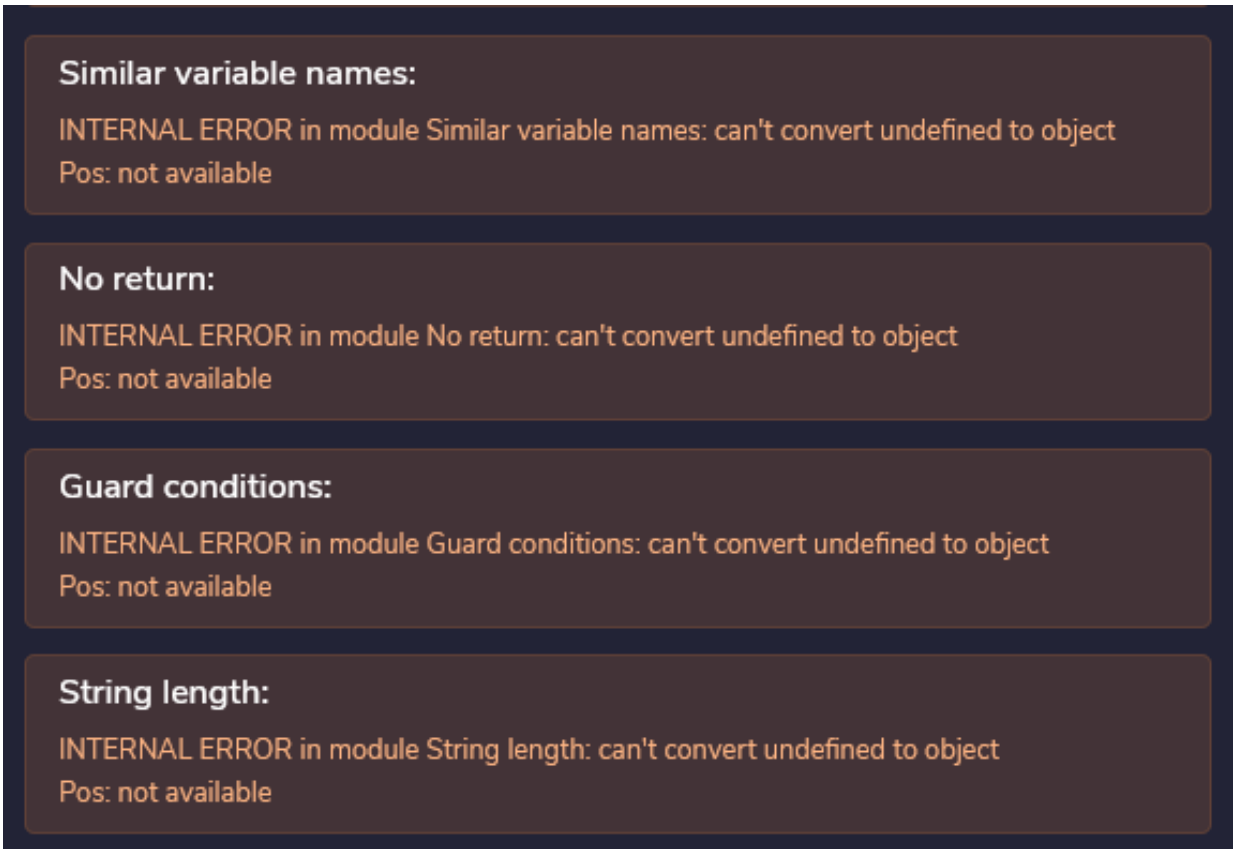
ERC20:

INTERNAL ERROR in module ERC20: can't convert undefined to object
Pos: not available

Miscellaneous

Constant/View/Pure functions:

INTERNAL ERROR in module Constant/View/Pure functions: can't convert undefined to object
Pos: not available



SOLHINT LINTER

```
CoinToken.sol:25:18: Error: Parse error: missing ';' at '{'
CoinToken.sol:33:18: Error: Parse error: missing ';' at '{'
CoinToken.sol:40:18: Error: Parse error: missing ';' at '{'
CoinToken.sol:52:18: Error: Parse error: missing ';' at '{'
CoinToken.sol:59:18: Error: Parse error: missing ';' at '{'
CoinToken.sol:89:18: Error: Parse error: missing ';' at '{'
CoinToken.sol:96:18: Error: Parse error: missing ';' at '{'
CoinToken.sol:103:18: Error: Parse error: missing ';' at '{'
```

Disclaimer

Quillhash audit is not a security warranty, investment advice, or an endorsement of the Hybrid Doge V2 CoinToken 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 Hybrid Doge Team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.

Closing Summary

Overall, smart contracts are very well written and adhere to guidelines.

No instances of Integer Overflow and Underflow vulnerabilities or Back-Door Entry were found in the contract, but relying on other contracts might cause Reentrancy Vulnerability.

QuillAudits

Canada, India, Singapore and United Kingdom

audits.quillhash.com

audits@quillhash.com