QuillAudits

Audit Report
February, 2022

For

QToken

Contents

Contents

Contents

Scope of the Audit

The scope of this audit was to analyze and document the QToken Token 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 analysed 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
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 analysed, 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, Theo.

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.

Risk-level	Description
High	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	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	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 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 issues per severity

Type	High	Medium	Low	Informational
Open	0	0	0	0
Acknowledged	0	1	2	0
Closed	1	0	2	6

Introduction

During the period of **January 24, 2021 to February 1, 2021** - QuillAudits Team performed a security audit for **QToken** smart contracts.

The code for the audit was taken from following the official link: https://github.com/qoneqt/Qtoken

V	Date	Commit ID	Files
1	22 Jan 2022	8efd2209fedc09a5530d 6e46a9df6a681c815031	1. QToken.sol 2. QTokenbuy.sol 3. QoneqtTokenGenerator.sol
2	30 Jan 2022	94b225a9ab5a7b10f742 ec5c4896f07a99d1a52d	1. QToken.sol 2. QTokenbuy.sol 3. QoneqtTokenGenerator.sol

Issues Found – Code Review / Manual Testing

A.Contract - QToken

High severity issues

No issues were found.

Medium severity issues

No issues were found.

Low severity issues

No issues were found.

Informational issues

No issues were found.

TestCases for Functional Testing

Some of the test cases which were part of the functional testing are listed below:

QToken.sol

- ERC20 transfer should work as expected **PASS**
- should be able to approve **PASS**
- should be able to return allowance of the approved address **PASS**
- Should revert if the sender’s balance is less than the amount **PASS**
- Should revert if the recipient’s balance and transfer amount’s sum is less than the recipient’s balance **PASS**
- Should revert if the caller don’t have the allowance for making a call to transfer function **PASS**
- should revert if the transfer amount is greater than the allowance **PASS**

B.Contract - QTokenBuy

High severity issues

B.1 uniRouter variable has been initialized to zero address

Line	Code

181	address public PANCAKESWAP_ROUTER_ADDRESS ;
	
	constructor(address _pancakeswap_router_address) {
	PANCAKESWAP_ROUTER_ADDRESS = _pancakeswap_router_address;
	manager = msg.sender;
	}
	
	modifier OnlyManager() {
	require(msg.sender == manager);
	_;
	}
	
	IPancakeRouter02 public uniRouter =
	IPancakeRouter02(PANCAKESWAP_ROUTER_ADDRESS);

Description
The variable uniRouter is initialized with the value of PANCAKESWAP_ROUTER_ADDRESS, at the time of construction. At that point of time PANCAKESWAP_ROUTER_ADDRESS is also zero address. Due to this swapETHForToken function will never work which uses the uniRouter to do swaps.

Remediation
Initialize uniRouter inside the constructor.

Status: Fixed

Medium severity issues

No issues were found.

Low severity issues

B.2 ERC20 transfers might revert for some tokens

Line	Code

201-203	function withdrawAllFromToken(IERC20 token) public OnlyManager {
	require(token.transfer(msg.sender, token.balanceOf(address(this))),
	"Transfer failed");
	}

Description
There are some ERC20 tokens that do not return anything on **transfer** or **transferFrom**. If this contract is ever used to interact with such tokens then, withdrawAllFromToken will revert.

Remediation
Use the Openzeppelin safeERC20 library to transfer ERC20 tokens, as it can handle all the scenarios.

Status: Fixed

Informational issues

B.3 State variables that could be declared immutable

Line	Code
166	address public manager;

Description
The above state variables should be declared immutable to save gas.

Remediation
Add the **immutable** attributes to state variables that never change after deployment.

Status: Fixed

B.4 Public function that could be declared external

Description

The following **public** functions that are never called by the contract should be declared **external** to save gas:

- withdrawAllFromToken
- withdrawWhatYouWantFromToken
- getBalanceOfToken

Remediation

Use the external attribute for functions never called from the contract.

Status: Fixed

TestCases for Functional Testing

Some of the test cases which were part of the functional testing are listed below:

QTokenbuy.sol

- Should be able to swap from ETH to desired token **PASS**
- Should be able to withdraw tokens from the contract **PASS**
- Only owner should be able to withdraw the tokens **PASS**

C.Contract - QoneqtTokenGenerator

High severity issues

No issues were found.

Medium severity issues

C.1 Centralization Risks

Description

The role owner has the authority to

- manage the list containing addresses excluded reward and **fee**.
- **withdraw ether** from the contract at any point of time.
- Change the fee rate

Remediation

We advise the client to handle the governance account carefully to avoid any potential hack. We also advise the client to consider the following solutions:

- with reasonable latency for community awareness on privileged operations;
- Multisig with community-voted 3rd-party independent co-signers;
- DAO or Governance module increasing transparency and community involvement;

Status: Acknowledged

Client's Comment

It's the user's choice to set the parameters accordingly that is the reason the token generator is built.

Low severity issues

C.2 Missing Range Check for Input Variable

Description

The owner can set the following state variables arbitrary large or small causing potential risks in fee deduction logic:

- _burnRate
- _liquidityFee

Remediation
We recommend setting ranges and check the above input variables.

Status: Acknowledged

Client's Comment
It's the user's choice to set the parameters accordingly that is the reason the token generator is built.

C.3 Missing zero address validation

Line	Code
449	448 constructor (string memory tokenName, string memory tokenSymbol,uint256 initi 449 address msgSender = _msgSender(); 450 _owner = msgSender; 451 _name = tokenName; 452 _symbol = tokenSymbol; 453 _tTotal = initialSupply * 10 ** 18; 454 _rTotal = (MAX - (MAX % _tTotal)); 455 LiqFee = _liqFee;
941	940 function transferXS(address payable recipient) public onlyOwner { 941 recipient.transfer(address(this).balance); 942 }

Description
There are missing zero checks in constructor and transferXS function and they could lead to unintended transfer of funds or in some cases loss of ownership.

Remediation
Use a "**require**" statement to check for zero address.

Status: Fixed

C.4 If ownable is set to false in constructor, then owner can never renounce ownership

Line	Code

494-497	function renounceOwnershipToBurnAddress() public virtual onlyOwner {
	require(Ownable == true , "Your token doesn't have Admin key burn option , It is decided at the time of token creation time");
	_owner = address(0);
	}

Description

If ownable is set to false in the constructor, then renounceOwnershipToBurnAddress function will always revert due the require check on the line **#495**.

Remediation

If this is not the intended logic, then remove that require check so that admin can renounce the ownership.

Status: Acknowledged

Client's Comment

It's the user's choice to set the parameters accordingly that is the reason the token generator is built.

Informational issues

C.5 Missing Events for Significant Transactions

Description

The missing event makes it difficult to track off-chain decimal changes. An event should be emitted for significant transactions calling the functions :

- excludeFromReward
- includeInReward
- includeInFee
- setLiquifyAmount
- excludeFromFee
- transferXS
- setBurnRate
- setLiquidityFeePercent
- transferOwnership

Remediation

We recommend emitting the appropriate events.

Status: Fixed

C.6 Incorrect error message

Line	Code
602	require(_isExcluded[account], "Account is already excluded");

Description

The above error messages do not describe the error correctly.

Remediation

Change the error messages to something that describes the error better.

Status: Fixed

C.7 Unused state variables

Line	Code
389	address private _previousOwner;

Description

The above state variables should be removed to improve code readability and save deployment gas.

Remediation

Remove the state variable.

Status: Fixed

C.8 Missing Events for Significant Transactions

Description

The following **public** functions that are never called by the contract should be declared **external** to save gas:

- isExcludedFromReward
- totalFees
- deliver
- reflectionFromToken
- excludeFromReward
- excludeFromFee
- includeInFee
- setBurnRate
- setSwapAndLiquifyEnabled
- setSwapMode
- burn
- isExcludedFromFee
- setLiquifyAmount
- withdrawAnyToken
- transferXS

Remediation

Use the external attribute for functions never called from the contract.

Status: Fixed

TestCases for Functional Testing

Some of the test cases which were part of the functional testing are listed below:

Qoneqttokengenerator.sol

- should be able to transfer tokens **PASS**
- should able to mint tokens **PASS**
- should be able to burn tokens **PASS**
- should be able to transfer ownership **PASS**
- should be able to approve **PASS**
- should be able to return allowance of the approved address **PASS**
- should revert if burn amount exceeds balance **PASS**
- Should revert if the sender’s balance is less than the amount **PASS**
- Should revert if the recipient’s balance and transfer amount’s sum is less than the recipient’s balance **PASS**
- Should revert if the caller don’t have the allowance for making a call to transfer function **PASS**
- should revert if the transfer amount is greater than the allowance **PASS**

Automated Tests

Slither

```
29/1, 5:10 PM

QtokenBuy.swapETHForToken(uint256,address[],uint256) (Qtokenbuy.sol#185-198) sends eth to arbitrary user
	Dangerous calls:
	- (success) = msg.sender.call{value: address(this).balance}() (Qtokenbuy.sol#196)
QoneqtTokenGenerator.addLiquidity(uint256,uint256) (QoneqtTokenGenerator.sol#864-877) sends eth to arbitrary user
	Dangerous calls:
	- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,address(this),block.timestamp) (QoneqtTokenGenerator.sol#869-876)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#functions-that-send-ether-to-arbitrary-destinations

IERC20 is re-used:
	- IERC20 (Qtokenbuy.sol#8-24)
	- IERC20 (QoneqtTokenGenerator.sol#6-26)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#name-reused

Reentrancy in QoneqtTokenGenerator._transfer(address,address,uint256) (QoneqtTokenGenerator.sol#776-821):
	External calls:
	- swapAndLiquify(contractTokenBalance) (QoneqtTokenGenerator.sol#800)
		- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,address(this),block.timestamp) (QoneqtTokenGenerator.sol#869-876)
		- uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (QoneqtTokenGenerator.sol#855-861)
	External calls sending eth:
	- swapAndLiquify(contractTokenBalance) (QoneqtTokenGenerator.sol#800)
		- uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,address(this),block.timestamp) (QoneqtTokenGenerator.sol#869-876)
	State variables written after the call(s):
	- _burn(from,tBurnAmount,rBurnAmount) (QoneqtTokenGenerator.sol#816)
		- _rOwned[address(sender)] = 0 (QoneqtTokenGenerator.sol#669)
		- _rOwned[address(sender)] -= rBurnAmount (QoneqtTokenGenerator.sol#671)
	- _tokenTransfer(from,to,amount,takeFee) (QoneqtTokenGenerator.sol#820)
		- _rOwned[address(this)] = _rOwned[address(this)].add(rLiquidity) (QoneqtTokenGenerator.sol#733)
		- _rOwned[sender] = _rOwned[sender].sub(rAmount) (QoneqtTokenGenerator.sol#902)
		- _rOwned[sender] = _rOwned[sender].sub(rAmount) (QoneqtTokenGenerator.sol#911)
		- _rOwned[sender] = _rOwned[sender].sub(rAmount) (QoneqtTokenGenerator.sol#922)
		- _rOwned[sender] = _rOwned[sender].sub(rAmount) (QoneqtTokenGenerator.sol#616)
		- _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (QoneqtTokenGenerator.sol#903)
		- _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (QoneqtTokenGenerator.sol#913)
		- _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (QoneqtTokenGenerator.sol#923)
		- _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (QoneqtTokenGenerator.sol#618)
	- _burn(from,tBurnAmount,rBurnAmount) (QoneqtTokenGenerator.sol#816)
		- _rTotal = _rTotal.sub(rBurnAmount) (QoneqtTokenGenerator.sol#674)
	- _tokenTransfer(from,to,amount,takeFee) (QoneqtTokenGenerator.sol#820)
		- _rTotal = _rTotal.sub(rFee) (QoneqtTokenGenerator.sol#656)
	- _tokenTransfer(from,to,amount,takeFee) (QoneqtTokenGenerator.sol#820)
		- _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (QoneqtTokenGenerator.sol#735)
		- _tOwned[sender] = _tOwned[sender].sub(tAmount) (QoneqtTokenGenerator.sol#921)
		- _tOwned[sender] = _tOwned[sender].sub(tAmount) (QoneqtTokenGenerator.sol#615)
		- _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (QoneqtTokenGenerator.sol#912)
		- _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (QoneqtTokenGenerator.sol#617)
	- _burn(from,tBurnAmount,rBurnAmount) (QoneqtTokenGenerator.sol#816)
		- _tTotal = _tTotal.sub(tBurnAmount) (QoneqtTokenGenerator.sol#673)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities

QtokenBuy.withdrawWhatYouWantFromToken(address,address,uint256) (Qtokenbuy.sol#206-209) ignores return value by IERC20(_tokenAddress).transfer(to,_amount) (Qtokenbuy.sol#207)
QoneqtTokenGenerator.withdrawAnyToken(address,address,uint256) (QoneqtTokenGenerator.sol#933-938) ignores return value by IERC20(_ERC20address).transfer(_recipient,_amount) (QoneqtTokenGenerator.sol#936)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-transfer
```

29/1, 5:10 PM

```
QoneqtTokenGenerator._getBurnAmounts(uint256) (QoneqtTokenGenerator.sol#660-665) performs a multiplication on the result of a division:
        -tBurnAmount = amount.mul(_burnRate).div(10 ** 2) (QoneqtTokenGenerator.sol#662)
        -rBurnAmount = tBurnAmount.mul(_currentRate) (QoneqtTokenGenerator.sol#663)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply

Reentrancy in QoneqtTokenGenerator.constructor(string,string,uint256,bool,bool,bool,uint256,uint256,address) (QoneqtTokenGenerator.sol#448-483):
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (QoneqtTokenGenerator.sol#477)
        State variables written after the call(s):
        - renounceOwnershipToBurnAddress() (QoneqtTokenGenerator.sol#480)
                - _owner = address(0) (QoneqtTokenGenerator.sol#496)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-1

QtokenBuy.swapETHForToken(uint256,address[],uint256) (Qtokenbuy.sol#185-198) ignores return value by uniRouter.swapExactETHForTokens{value: msg.value}(amountoutmin,pair,address(this),deadline) (Qtokenbuy.sol#187-19
2)
QoneqtTokenGenerator.addLiquidity(uint256,uint256) (QoneqtTokenGenerator.sol#864-877) ignores return value by uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,address(this),block.time
stamp) (QoneqtTokenGenerator.sol#869-876)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

QoneqtTokenGenerator.transferOwnership(address) (QoneqtTokenGenerator.sol#500-504) should emit an event for:
        - _owner = newOwner (QoneqtTokenGenerator.sol#503)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-access-control

QoneqtTokenGenerator.setLiquidityFeePercent(uint256) (QoneqtTokenGenerator.sol#632-636) should emit an event for:
        - _liquidityFee = liquidityFee (QoneqtTokenGenerator.sol#634)
QoneqtTokenGenerator.setBurnRate(uint256) (QoneqtTokenGenerator.sol#638-641) should emit an event for:
        - _burnRate = amount (QoneqtTokenGenerator.sol#640)
QoneqtTokenGenerator.setLiquifyAmount(uint256) (QoneqtTokenGenerator.sol#929-931) should emit an event for:
        - numTokensSellToAddToLiquidity = amount (QoneqtTokenGenerator.sol#930)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic

QtokenBuy.constructor(address)._pancakeswap_router_address (Qtokenbuy.sol#171) lacks a zero-check on :
                - PANCAKESWAP_ROUTER_ADDRESS = _pancakeswap_router_address (Qtokenbuy.sol#172)
QoneqtTokenGenerator.constructor(string,string,uint256,bool,bool,bool,uint256,uint256,address).msgSender (QoneqtTokenGenerator.sol#449) lacks a zero-check on :
                - _owner = msgSender (QoneqtTokenGenerator.sol#450)
QoneqtTokenGenerator.constructor(string,string,uint256,bool,bool,bool,uint256,uint256,address)._uniswapv2router (QoneqtTokenGenerator.sol#448) lacks a zero-check on :
                - uniswapv2routeraddress = _uniswapv2router (QoneqtTokenGenerator.sol#474)
QoneqtTokenGenerator.transferXS(address).recipient (QoneqtTokenGenerator.sol#940) lacks a zero-check on :
                - recipient.transfer(address(this).balance) (QoneqtTokenGenerator.sol#941)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

Reentrancy in QoneqtTokenGenerator._transfer(address,address,uint256) (QoneqtTokenGenerator.sol#776-821):
        External calls:
        - swapAndLiquify(contractTokenBalance) (QoneqtTokenGenerator.sol#800)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,address(this),block.timestamp) (QoneqtTokenGenerator.sol#869-876)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (QoneqtTokenGenerator.sol#855-861)
        External calls sending eth:
        - swapAndLiquify(contractTokenBalance) (QoneqtTokenGenerator.sol#800)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,address(this),block.timestamp) (QoneqtTokenGenerator.sol#869-876)
        State variables written after the call(s):
        - _tokenTransfer(from,to,amount,takeFee) (QoneqtTokenGenerator.sol#820)
                - _liquidityFee = _previousLiquidityFee (QoneqtTokenGenerator.sol#761)
                - _liquidityFee = 0 (QoneqtTokenGenerator.sol#756)
        - _tokenTransfer(from,to,amount,takeFee) (QoneqtTokenGenerator.sol#820)
```

29/1, 5:10 PM

```
        - _tokenTransfer(from,to,amount,takeFee) (QoneqtTokenGenerator.sol#820)
                - _taxFee = _previousTaxFee (QoneqtTokenGenerator.sol#760)
                - _taxFee = 0 (QoneqtTokenGenerator.sol#755)
        - _burn(from,tBurnAmount,rBurnAmount) (QoneqtTokenGenerator.sol#816)
                - _totalBurned = _totalBurned.add(tBurnAmount) (QoneqtTokenGenerator.sol#675)
Reentrancy in QoneqtTokenGenerator.constructor(string,string,uint256,bool,bool,bool,uint256,uint256,address) (QoneqtTokenGenerator.sol#448-483):
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (QoneqtTokenGenerator.sol#477)
        State variables written after the call(s):
        - uniswapV2Router = _uniswapV2Router (QoneqtTokenGenerator.sol#478)
Reentrancy in QoneqtTokenGenerator.swapAndLiquify(uint256) (QoneqtTokenGenerator.sol#823-844):
        External calls:
        - swapTokensForEth(half) (QoneqtTokenGenerator.sol#835)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (QoneqtTokenGenerator.sol#855-861)
        - addLiquidity(otherHalf,newBalance) (QoneqtTokenGenerator.sol#841)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,address(this),block.timestamp) (QoneqtTokenGenerator.sol#869-876)
        External calls sending eth:
        - addLiquidity(otherHalf,newBalance) (QoneqtTokenGenerator.sol#841)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,address(this),block.timestamp) (QoneqtTokenGenerator.sol#869-876)
        State variables written after the call(s):
        - addLiquidity(otherHalf,newBalance) (QoneqtTokenGenerator.sol#841)
                - _allowances[owner][spender] = amount (QoneqtTokenGenerator.sol#772)
Reentrancy in QoneqtTokenGenerator.transferFrom(address,address,uint256) (QoneqtTokenGenerator.sol#541-545):
        External calls:
        - _transfer(sender,recipient,amount) (QoneqtTokenGenerator.sol#542)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,address(this),block.timestamp) (QoneqtTokenGenerator.sol#869-876)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (QoneqtTokenGenerator.sol#855-861)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (QoneqtTokenGenerator.sol#542)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,address(this),block.timestamp) (QoneqtTokenGenerator.sol#869-876)
        State variables written after the call(s):
        - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (QoneqtTokenGenerator.sol#543)
                - _allowances[owner][spender] = amount (QoneqtTokenGenerator.sol#772)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2

Reentrancy in QoneqtTokenGenerator._transfer(address,address,uint256) (QoneqtTokenGenerator.sol#776-821):
        External calls:
        - swapAndLiquify(contractTokenBalance) (QoneqtTokenGenerator.sol#800)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,address(this),block.timestamp) (QoneqtTokenGenerator.sol#869-876)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (QoneqtTokenGenerator.sol#855-861)
        External calls sending eth:
        - swapAndLiquify(contractTokenBalance) (QoneqtTokenGenerator.sol#800)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,address(this),block.timestamp) (QoneqtTokenGenerator.sol#869-876)
        Event emitted after the call(s):
        - Transfer(sender,address(0),tBurnAmount) (QoneqtTokenGenerator.sol#677)
                - _burn(from,tBurnAmount,rBurnAmount) (QoneqtTokenGenerator.sol#816)
        - Transfer(sender,recipient,tTransferAmount) (QoneqtTokenGenerator.sol#906)
                - _tokenTransfer(from,to,amount,takeFee) (QoneqtTokenGenerator.sol#820)
        - Transfer(sender,recipient,tTransferAmount) (QoneqtTokenGenerator.sol#926)
                - _tokenTransfer(from,to,amount,takeFee) (QoneqtTokenGenerator.sol#820)
        - Transfer(sender,recipient,tTransferAmount) (QoneqtTokenGenerator.sol#916)
                - _tokenTransfer(from,to,amount,takeFee) (QoneqtTokenGenerator.sol#820)
        - Transfer(sender,recipient,tTransferAmount) (QoneqtTokenGenerator.sol#621)
                - _tokenTransfer(from,to,amount,takeFee) (QoneqtTokenGenerator.sol#820)
Reentrancy in QoneqtTokenGenerator.swapAndLiquify(uint256) (QoneqtTokenGenerator.sol#823-844):
```

```
29/1, 5:11 PM
        External calls:
        - swapTokensForEth(half) (QoneqtTokenGenerator.sol#835)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (QoneqtTokenGenerator.sol#855-861)
        - addLiquidity(otherHalf,newBalance) (QoneqtTokenGenerator.sol#841)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,address(this),block.timestamp) (QoneqtTokenGenerator.sol#869-876)
        External calls sending eth:
        - addLiquidity(otherHalf,newBalance) (QoneqtTokenGenerator.sol#841)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,address(this),block.timestamp) (QoneqtTokenGenerator.sol#869-876)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (QoneqtTokenGenerator.sol#773)
                - addLiquidity(otherHalf,newBalance) (QoneqtTokenGenerator.sol#841)
        - SwapAndLiquify(half,newBalance,otherHalf) (QoneqtTokenGenerator.sol#843)
Reentrancy in QoneqtTokenGenerator.transferFrom(address,address,uint256) (QoneqtTokenGenerator.sol#541-545):
        External calls:
        - _transfer(sender,recipient,amount) (QoneqtTokenGenerator.sol#542)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,address(this),block.timestamp) (QoneqtTokenGenerator.sol#869-876)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (QoneqtTokenGenerator.sol#855-861)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (QoneqtTokenGenerator.sol#542)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,address(this),block.timestamp) (QoneqtTokenGenerator.sol#869-876)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (QoneqtTokenGenerator.sol#773)
                - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (QoneqtTokenGenerator.sol#543)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Address.isContract(address) (QoneqtTokenGenerator.sol#104-113) uses assembly
        - INLINE ASM (QoneqtTokenGenerator.sol#111)
Address._functionCallWithValue(address,bytes,uint256,string) (QoneqtTokenGenerator.sol#149-170) uses assembly
        - INLINE ASM (QoneqtTokenGenerator.sol#162-165)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

QoneqtTokenGenerator.constructor(string,string,uint256,bool,bool,bool,uint256,uint256,address) (QoneqtTokenGenerator.sol#448-483) compares to a boolean constant:
        -_burnFee == true (QoneqtTokenGenerator.sol#463)
QoneqtTokenGenerator.constructor(string,string,uint256,bool,bool,bool,uint256,uint256,address) (QoneqtTokenGenerator.sol#448-483) compares to a boolean constant:
        -Ownable == true (QoneqtTokenGenerator.sol#479)
QoneqtTokenGenerator.constructor(string,string,uint256,bool,bool,bool,uint256,uint256,address) (QoneqtTokenGenerator.sol#448-483) compares to a boolean constant:
        -_liqFee == true (QoneqtTokenGenerator.sol#458)
QoneqtTokenGenerator.constructor(string,string,uint256,bool,bool,bool,uint256,uint256,address) (QoneqtTokenGenerator.sol#448-483) compares to a boolean constant:
        -_liqFee == false (QoneqtTokenGenerator.sol#460)
QoneqtTokenGenerator.constructor(string,string,uint256,bool,bool,bool,uint256,uint256,address) (QoneqtTokenGenerator.sol#448-483) compares to a boolean constant:
        -_burnFee == false (QoneqtTokenGenerator.sol#465)
QoneqtTokenGenerator.renounceOwnershipToBurnAddress() (QoneqtTokenGenerator.sol#494-497) compares to a boolean constant:
        -require(bool,string)(Ownable == true,Your token doesn't have Admin key burn option , It is decided at the time of token creation time) (QoneqtTokenGenerator.sol#495)
QoneqtTokenGenerator.setLiquidityFeePercent(uint256) (QoneqtTokenGenerator.sol#632-636) compares to a boolean constant:
        -require(bool,string)(LiqFee == true,Liquidity already fixed) (QoneqtTokenGenerator.sol#633)
QoneqtTokenGenerator.setBurnRate(uint256) (QoneqtTokenGenerator.sol#638-641) compares to a boolean constant:
        -require(bool,string)(burnFee == true,BurnFee already fixed) (QoneqtTokenGenerator.sol#639)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#boolean-equality

Different versions of Solidity is used:
        - Version used: ['>=0.4.22<0.9.0', '^0.8.0']
        - ^0.8.0 (Qtokenbuy.sol#4)
        - ^0.8.0 (QoneqtTokenGenerator.sol#4)
        - >=0.4.22<0.9.0 (Migrations.sol#2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used
```

```
29/1, 5:11 PM
        - _excluded.pop() (QoneqtTokenGenerator.sol#608)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop

Address._functionCallWithValue(address,bytes,uint256,string) (QoneqtTokenGenerator.sol#149-170) is never used and should be removed
Address.functionCall(address,bytes) (QoneqtTokenGenerator.sol#125-127) is never used and should be removed
Address.functionCall(address,bytes,string) (QoneqtTokenGenerator.sol#135-137) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (QoneqtTokenGenerator.sol#140-142) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (QoneqtTokenGenerator.sol#144-147) is never used and should be removed
Address.isContract(address) (QoneqtTokenGenerator.sol#104-113) is never used and should be removed
Address.sendValue(address,uint256) (QoneqtTokenGenerator.sol#116-122) is never used and should be removed
Context._msgData() (QoneqtTokenGenerator.sol#95-98) is never used and should be removed
SafeMath.mod(uint256,uint256) (QoneqtTokenGenerator.sol#79-81) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (QoneqtTokenGenerator.sol#84-87) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

QtokenBuy.uniRouter (Qtokenbuy.sol#181) is set pre-construction with a non-constant function or state variable:
        - IPancakeRouter02(PANCAKESWAP_ROUTER_ADDRESS)
QoneqtTokenGenerator._previousTaxFee (QoneqtTokenGenerator.sol#411) is set pre-construction with a non-constant function or state variable:
        - _taxFee
QoneqtTokenGenerator._previousLiquidityFee (QoneqtTokenGenerator.sol#414) is set pre-construction with a non-constant function or state variable:
        - _liquidityFee
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#function-initializing-state

Pragma version^0.8.0 (Qtokenbuy.sol#4) allows old versions
Pragma version^0.8.0 (QoneqtTokenGenerator.sol#4) allows old versions
Pragma version>=0.4.22<0.9.0 (Migrations.sol#2) is too complex
solc-0.8.2 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in QtokenBuy.swapETHForToken(uint256,address[],uint256) (Qtokenbuy.sol#185-198):
        - (success) = msg.sender.call{value: address(this).balance}() (Qtokenbuy.sol#196)
Low level call in Address.sendValue(address,uint256) (QoneqtTokenGenerator.sol#116-122):
        - (success) = recipient.call{value: amount}() (QoneqtTokenGenerator.sol#120)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (QoneqtTokenGenerator.sol#149-170):
        - (success,returndata) = target.call{value: weiValue}(data) (QoneqtTokenGenerator.sol#153)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function IPancakeRouter01.WETH() (Qtokenbuy.sol#28) is not in mixedCase
Parameter QtokenBuy.withdrawWhatYouWantFromToken(address,address,uint256)._tokenAddress (Qtokenbuy.sol#206) is not in mixedCase
Parameter QtokenBuy.withdrawWhatYouWantFromToken(address,address,uint256)._amount (Qtokenbuy.sol#206) is not in mixedCase
Parameter QtokenBuy.getBalanceOfToken(address)._token (Qtokenbuy.sol#212) is not in mixedCase
Variable QtokenBuy.PANCAKESWAP_ROUTER_ADDRESS (Qtokenbuy.sol#168) is not in mixedCase
Modifier QtokenBuy.OnlyManager() (Qtokenbuy.sol#176-179) is not in mixedCase
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (QoneqtTokenGenerator.sol#207) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (QoneqtTokenGenerator.sol#208) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (QoneqtTokenGenerator.sol#225) is not in mixedCase
Function IUniswapV2Router01.WETH() (QoneqtTokenGenerator.sol#247) is not in mixedCase
Function QoneqtTokenGenerator.Admin() (QoneqtTokenGenerator.sol#484-486) is not in mixedCase
Parameter QoneqtTokenGenerator.setSwapAndLiquifyEnabled(bool)._enabled (QoneqtTokenGenerator.sol#643) is not in mixedCase
Parameter QoneqtTokenGenerator.calculateTaxFee(uint256)._amount (QoneqtTokenGenerator.sol#738) is not in mixedCase
Parameter QoneqtTokenGenerator.calculateLiquidityFee(uint256)._amount (QoneqtTokenGenerator.sol#744) is not in mixedCase
Parameter QoneqtTokenGenerator.withdrawAnyToken(address,address,uint256)._recipient (QoneqtTokenGenerator.sol#933) is not in mixedCase
Parameter QoneqtTokenGenerator.withdrawAnyToken(address,address,uint256)._ERC20address (QoneqtTokenGenerator.sol#933) is not in mixedCase
Parameter QoneqtTokenGenerator.withdrawAnyToken(address,address,uint256)._amount (QoneqtTokenGenerator.sol#933) is not in mixedCase
Variable QoneqtTokenGenerator._taxFee (QoneqtTokenGenerator.sol#410) is not in mixedCase
```

29/1, 5:11 PM

```
Variable QoneqtTokenGenerator._liquidityFee (QoneqtTokenGenerator.sol#413) is not in mixedCase
Variable QoneqtTokenGenerator._burnRate (QoneqtTokenGenerator.sol#416) is not in mixedCase
Variable QoneqtTokenGenerator._totalBurned (QoneqtTokenGenerator.sol#417) is not in mixedCase
Variable QoneqtTokenGenerator.LiqFee (QoneqtTokenGenerator.sol#419) is not in mixedCase
Variable QoneqtTokenGenerator.Ownable (QoneqtTokenGenerator.sol#421) is not in mixedCase
Variable QoneqtTokenGenerator._swapMode (QoneqtTokenGenerator.sol#429) is not in mixedCase
Variable Migrations.last_completed_migration (Migrations.sol#6) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (QoneqtTokenGenerator.sol#96)" inContext (QoneqtTokenGenerator.sol#90-99)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

Variable IPancakeRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (Qtokenbuy.sol#33) is too similar to IPancakeRouter01.addLiquidity(address,address,uint256,uint
256,uint256,uint256,address,uint256).amountBDesired (Qtokenbuy.sol#34)
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (QoneqtTokenGenerator.sol#252) is too similar to IUniswapV2Router01.addLiquidity(address,addr
ess,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (QoneqtTokenGenerator.sol#253)
Variable QoneqtTokenGenerator._liquidityFee (QoneqtTokenGenerator.sol#413) is too similar to QoneqtTokenGenerator.constructor(string,string,uint256,bool,bool,bool,uint256,uint256,address).liquidityFee_ (QoneqtToken
Generator.sol#448)
Variable QoneqtTokenGenerator._burn(address,uint256,uint256).rBurnAmount (QoneqtTokenGenerator.sol#667) is too similar to QoneqtTokenGenerator.burn(uint256).tBurnAmount (QoneqtTokenGenerator.sol#685)
Variable QoneqtTokenGenerator._getBurnAmounts(uint256).rBurnAmount (QoneqtTokenGenerator.sol#663) is too similar to QoneqtTokenGenerator._transfer(address,address,uint256).tBurnAmount (QoneqtTokenGenerator.sol#814)
Variable QoneqtTokenGenerator.burn(uint256).rBurnAmount (QoneqtTokenGenerator.sol#686) is too similar to QoneqtTokenGenerator._transfer(address,address,uint256).tBurnAmount (QoneqtTokenGenerator.sol#814)
Variable QoneqtTokenGenerator.burn(uint256).rBurnAmount (QoneqtTokenGenerator.sol#686) is too similar to QoneqtTokenGenerator._getBurnAmounts(uint256).tBurnAmount (QoneqtTokenGenerator.sol#662)
Variable QoneqtTokenGenerator._burn(address,uint256,uint256).rBurnAmount (QoneqtTokenGenerator.sol#667) is too similar to QoneqtTokenGenerator._transfer(address,address,uint256).tBurnAmount (QoneqtTokenGenerator.so
l#814)
Variable QoneqtTokenGenerator._burn(address,uint256,uint256).rBurnAmount (QoneqtTokenGenerator.sol#667) is too similar to QoneqtTokenGenerator._getBurnAmounts(uint256).tBurnAmount (QoneqtTokenGenerator.sol#662)
Variable QoneqtTokenGenerator._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (QoneqtTokenGenerator.sol#709) is too similar to QoneqtTokenGenerator._transferBothExcluded(address,address,uint256).tTrans
ferAmount (QoneqtTokenGenerator.sol#614)
Variable QoneqtTokenGenerator._transferStandard(address,address,uint256).rTransferAmount (QoneqtTokenGenerator.sol#901) is too similar to QoneqtTokenGenerator._transferStandard(address,address,uint256).tTransferAmo
unt (QoneqtTokenGenerator.sol#901)
Variable QoneqtTokenGenerator._getValues(uint256).rTransferAmount (QoneqtTokenGenerator.sol#694) is too similar to QoneqtTokenGenerator._transferBothExcluded(address,address,uint256).tTransferAmount (QoneqtTokenGen
erator.sol#614)
Variable QoneqtTokenGenerator._transferBothExcluded(address,address,uint256).rTransferAmount (QoneqtTokenGenerator.sol#614) is too similar to QoneqtTokenGenerator._transferBothExcluded(address,address,uint256).tTra
nsferAmount (QoneqtTokenGenerator.sol#614)
Variable QoneqtTokenGenerator.reflectionFromToken(uint256,bool).rTransferAmount (QoneqtTokenGenerator.sol#580) is too similar to QoneqtTokenGenerator._transferToExcluded(address,address,uint256).tTransferAmount (Qo
neqtTokenGenerator.sol#910)
Variable QoneqtTokenGenerator.reflectionFromToken(uint256,bool).rTransferAmount (QoneqtTokenGenerator.sol#580) is too similar to QoneqtTokenGenerator._getValues(uint256).tTransferAmount (QoneqtTokenGenerator.sol#69
3)
Variable QoneqtTokenGenerator._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (QoneqtTokenGenerator.sol#709) is too similar to QoneqtTokenGenerator._transferFromExcluded(address,address,uint256).tTrans
ferAmount (QoneqtTokenGenerator.sol#920)
Variable QoneqtTokenGenerator.reflectionFromToken(uint256,bool).rTransferAmount (QoneqtTokenGenerator.sol#580) is too similar to QoneqtTokenGenerator._transferStandard(address,address,uint256).tTransferAmount (Qone
qtTokenGenerator.sol#901)
Variable QoneqtTokenGenerator._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (QoneqtTokenGenerator.sol#709) is too similar to QoneqtTokenGenerator._getTValues(uint256).tTransferAmount (QoneqtTokenGene
rator.sol#701)
Variable QoneqtTokenGenerator._getValues(uint256).rTransferAmount (QoneqtTokenGenerator.sol#694) is too similar to QoneqtTokenGenerator._transferFromExcluded(address,address,uint256).tTransferAmount (QoneqtTokenGen
erator.sol#920)
Variable QoneqtTokenGenerator._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (QoneqtTokenGenerator.sol#709) is too similar to QoneqtTokenGenerator._transferToExcluded(address,address,uint256).tTransfe
rAmount (QoneqtTokenGenerator.sol#910)
Variable QoneqtTokenGenerator._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (QoneqtTokenGenerator.sol#709) is too similar to QoneqtTokenGenerator._getValues(uint256).tTransferAmount (QoneqtTokenGener
ator.sol#693)
Variable QoneqtTokenGenerator._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (QoneqtTokenGenerator.sol#709) is too similar to QoneqtTokenGenerator._transferStandard(address,address,uint256).tTransferA
mount (QoneqtTokenGenerator.sol#901)
Variable QoneqtTokenGenerator._getValues(uint256).rTransferAmount (QoneqtTokenGenerator.sol#694) is too similar to QoneqtTokenGenerator._getTValues(uint256).tTransferAmount (QoneqtTokenGenerator.sol#701)
Variable QoneqtTokenGenerator._getValues(uint256).rTransferAmount (QoneqtTokenGenerator.sol#694) is too similar to QoneqtTokenGenerator._transferToExcluded(address,address,uint256).tTransferAmount (QoneqtTokenGener
ator.sol#910)
Variable QoneqtTokenGenerator._getValues(uint256).rTransferAmount (QoneqtTokenGenerator.sol#694) is too similar to QoneqtTokenGenerator._transferStandard(address,address,uint256).tTransferAmount (QoneqtTokenGenerat
```

29/1, 5:11 PM

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar

QoneqtTokenGenerator._previousOwner (QoneqtTokenGenerator.sol#389) is never used in QoneqtTokenGenerator (QoneqtTokenGenerator.sol#384-945)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

QoneqtTokenGenerator._decimals (QoneqtTokenGenerator.sol#408) should be constant
QoneqtTokenGenerator._previousOwner (QoneqtTokenGenerator.sol#389) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

withdrawAllFromToken(IERC20) should be declared external:
        - QtokenBuy.withdrawAllFromToken(IERC20) (Qtokenbuy.sol#201-203)
withdrawWhatYouWantFromToken(address,address,uint256) should be declared external:
        - QtokenBuy.withdrawWhatYouWantFromToken(address,address,uint256) (Qtokenbuy.sol#206-209)
getBalanceOfToken(address) should be declared external:
        - QtokenBuy.getBalanceOfToken(address) (Qtokenbuy.sol#212-214)
transferOwnership(address) should be declared external:
        - QoneqtTokenGenerator.transferOwnership(address) (QoneqtTokenGenerator.sol#500-504)
name() should be declared external:
        - QoneqtTokenGenerator.name() (QoneqtTokenGenerator.sol#506-508)
symbol() should be declared external:
        - QoneqtTokenGenerator.symbol() (QoneqtTokenGenerator.sol#510-512)
decimals() should be declared external:
        - QoneqtTokenGenerator.decimals() (QoneqtTokenGenerator.sol#514-516)
totalSupply() should be declared external:
        - QoneqtTokenGenerator.totalSupply() (QoneqtTokenGenerator.sol#518-520)
transfer(address,uint256) should be declared external:
        - QoneqtTokenGenerator.transfer(address,uint256) (QoneqtTokenGenerator.sol#527-530)
allowance(address,address) should be declared external:
        - QoneqtTokenGenerator.allowance(address,address) (QoneqtTokenGenerator.sol#532-534)
approve(address,uint256) should be declared external:
        - QoneqtTokenGenerator.approve(address,uint256) (QoneqtTokenGenerator.sol#536-539)
transferFrom(address,address,uint256) should be declared external:
        - QoneqtTokenGenerator.transferFrom(address,address,uint256) (QoneqtTokenGenerator.sol#541-545)
increaseAllowance(address,uint256) should be declared external:
        - QoneqtTokenGenerator.increaseAllowance(address,uint256) (QoneqtTokenGenerator.sol#547-550)
decreaseAllowance(address,uint256) should be declared external:
        - QoneqtTokenGenerator.decreaseAllowance(address,uint256) (QoneqtTokenGenerator.sol#552-555)
isExcludedFromReward(address) should be declared external:
        - QoneqtTokenGenerator.isExcludedFromReward(address) (QoneqtTokenGenerator.sol#557-559)
totalFees() should be declared external:
        - QoneqtTokenGenerator.totalFees() (QoneqtTokenGenerator.sol#561-563)
deliver(uint256) should be declared external:
        - QoneqtTokenGenerator.deliver(uint256) (QoneqtTokenGenerator.sol#565-572)
reflectionFromToken(uint256,bool) should be declared external:
        - QoneqtTokenGenerator.reflectionFromToken(uint256,bool) (QoneqtTokenGenerator.sol#574-583)
excludeFromReward(address) should be declared external:
        - QoneqtTokenGenerator.excludeFromReward(address) (QoneqtTokenGenerator.sol#591-599)
excludeFromFee(address) should be declared external:
        - QoneqtTokenGenerator.excludeFromFee(address) (QoneqtTokenGenerator.sol#624-626)
includeInFee(address) should be declared external:
        - QoneqtTokenGenerator.includeInFee(address) (QoneqtTokenGenerator.sol#628-630)
setBurnRate(uint256) should be declared external:
        - QoneqtTokenGenerator.setBurnRate(uint256) (QoneqtTokenGenerator.sol#638-641)
setSwapAndLiquifyEnabled(bool) should be declared external:
        - QoneqtTokenGenerator.setSwapAndLiquifyEnabled(bool) (QoneqtTokenGenerator.sol#643-646)
```

```
29/1, 5:11 PM
name() should be declared external:
        - QoneqtTokenGenerator.name() (QoneqtTokenGenerator.sol#506-508)
symbol() should be declared external:
        - QoneqtTokenGenerator.symbol() (QoneqtTokenGenerator.sol#510-512)
decimals() should be declared external:
        - QoneqtTokenGenerator.decimals() (QoneqtTokenGenerator.sol#514-516)
totalSupply() should be declared external:
        - QoneqtTokenGenerator.totalSupply() (QoneqtTokenGenerator.sol#518-520)
transfer(address,uint256) should be declared external:
        - QoneqtTokenGenerator.transfer(address,uint256) (QoneqtTokenGenerator.sol#527-530)
allowance(address,address) should be declared external:
        - QoneqtTokenGenerator.allowance(address,address) (QoneqtTokenGenerator.sol#532-534)
approve(address,uint256) should be declared external:
        - QoneqtTokenGenerator.approve(address,uint256) (QoneqtTokenGenerator.sol#536-539)
transferFrom(address,address,uint256) should be declared external:
        - QoneqtTokenGenerator.transferFrom(address,address,uint256) (QoneqtTokenGenerator.sol#541-545)
increaseAllowance(address,uint256) should be declared external:
        - QoneqtTokenGenerator.increaseAllowance(address,uint256) (QoneqtTokenGenerator.sol#547-550)
decreaseAllowance(address,uint256) should be declared external:
        - QoneqtTokenGenerator.decreaseAllowance(address,uint256) (QoneqtTokenGenerator.sol#552-555)
isExcludedFromReward(address) should be declared external:
        - QoneqtTokenGenerator.isExcludedFromReward(address) (QoneqtTokenGenerator.sol#557-559)
totalFees() should be declared external:
        - QoneqtTokenGenerator.totalFees() (QoneqtTokenGenerator.sol#561-563)
deliver(uint256) should be declared external:
        - QoneqtTokenGenerator.deliver(uint256) (QoneqtTokenGenerator.sol#565-572)
reflectionFromToken(uint256,bool) should be declared external:
        - QoneqtTokenGenerator.reflectionFromToken(uint256,bool) (QoneqtTokenGenerator.sol#574-583)
excludeFromReward(address) should be declared external:
        - QoneqtTokenGenerator.excludeFromReward(address) (QoneqtTokenGenerator.sol#591-599)
excludeFromFee(address) should be declared external:
        - QoneqtTokenGenerator.excludeFromFee(address) (QoneqtTokenGenerator.sol#624-626)
includeInFee(address) should be declared external:
        - QoneqtTokenGenerator.includeInFee(address) (QoneqtTokenGenerator.sol#628-630)
setBurnRate(uint256) should be declared external:
        - QoneqtTokenGenerator.setBurnRate(uint256) (QoneqtTokenGenerator.sol#638-641)
setSwapAndLiquifyEnabled(bool) should be declared external:
        - QoneqtTokenGenerator.setSwapAndLiquifyEnabled(bool) (QoneqtTokenGenerator.sol#643-646)
setSwapMode(bool) should be declared external:
        - QoneqtTokenGenerator.setSwapMode(bool) (QoneqtTokenGenerator.sol#648-650)
burn(uint256) should be declared external:
        - QoneqtTokenGenerator.burn(uint256) (QoneqtTokenGenerator.sol#680-690)
isExcludedFromFee(address) should be declared external:
        - QoneqtTokenGenerator.isExcludedFromFee(address) (QoneqtTokenGenerator.sol#764-766)
setLiquifyAmount(uint256) should be declared external:
        - QoneqtTokenGenerator.setLiquifyAmount(uint256) (QoneqtTokenGenerator.sol#929-931)
withdrawAnyToken(address,address,uint256) should be declared external:
        - QoneqtTokenGenerator.withdrawAnyToken(address,address,uint256) (QoneqtTokenGenerator.sol#933-938)
transferXS(address) should be declared external:
        - QoneqtTokenGenerator.transferXS(address) (QoneqtTokenGenerator.sol#940-942)
setCompleted(uint256) should be declared external:
        - Migrations.setCompleted(uint256) (Migrations.sol#16-18)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
. analyzed (14 contracts with 77 detectors), 177 result(s) found
```

```
SafeMath.div(uint256,uint256) (Qtoken.sol#12-17) is never used and should be removed
SafeMath.mul(uint256,uint256) (Qtoken.sol#6-10) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version>=0.4.22<0.9.0 (Migrations.sol#2) is too complex
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Parameter Qtoken.balanceOf(address)._owner (Qtoken.sol#64) is not in mixedCase
Parameter Qtoken.transfer(address,uint256)._to (Qtoken.sol#69) is not in mixedCase
Parameter Qtoken.transfer(address,uint256)._value (Qtoken.sol#69) is not in mixedCase
Parameter Qtoken.transferFrom(address,address,uint256)._from (Qtoken.sol#81) is not in mixedCase
Parameter Qtoken.transferFrom(address,address,uint256)._to (Qtoken.sol#81) is not in mixedCase
Parameter Qtoken.transferFrom(address,address,uint256)._value (Qtoken.sol#81) is not in mixedCase
Parameter Qtoken.approve(address,uint256)._spender (Qtoken.sol#98) is not in mixedCase
Parameter Qtoken.approve(address,uint256)._value (Qtoken.sol#98) is not in mixedCase
Parameter Qtoken.allowance(address,address)._owner (Qtoken.sol#105) is not in mixedCase
Parameter Qtoken.allowance(address,address)._spender (Qtoken.sol#105) is not in mixedCase
Parameter Qtoken.increaseApproval(address,uint256)._spender (Qtoken.sol#110) is not in mixedCase
Parameter Qtoken.increaseApproval(address,uint256)._addedValue (Qtoken.sol#110) is not in mixedCase
Parameter Qtoken.decreaseApproval(address,uint256)._spender (Qtoken.sol#117) is not in mixedCase
Parameter Qtoken.decreaseApproval(address,uint256)._subtractedValue (Qtoken.sol#117) is not in mixedCase
Constant Qtoken.totalSupply (Qtoken.sol#46) is not in UPPER_CASE_WITH_UNDERSCORES
Variable Migrations.last_completed_migration (Migrations.sol#6) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Qtoken.slitherConstructorConstantVariables() (Qtoken.sol#32-128) uses literals with too many digits:
        - totalSupply = 100000000 * (10 ** uint256(decimals)) (Qtoken.sol#46)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

balanceOf(address) should be declared external:
        - Qtoken.balanceOf(address) (Qtoken.sol#64-66)
transfer(address,uint256) should be declared external:
        - Qtoken.transfer(address,uint256) (Qtoken.sol#69-78)
transferFrom(address,address,uint256) should be declared external:
        - Qtoken.transferFrom(address,address,uint256) (Qtoken.sol#81-91)
approve(address,uint256) should be declared external:
        - Qtoken.approve(address,uint256) (Qtoken.sol#98-102)
allowance(address,address) should be declared external:
        - Qtoken.allowance(address,address) (Qtoken.sol#105-107)
increaseApproval(address,uint256) should be declared external:
        - Qtoken.increaseApproval(address,uint256) (Qtoken.sol#110-114)
decreaseApproval(address,uint256) should be declared external:
        - Qtoken.decreaseApproval(address,uint256) (Qtoken.sol#117-126)
setCompleted(uint256) should be declared external:
        - Migrations.setCompleted(uint256) (Migrations.sol#16-18)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
. analyzed (3 contracts with 77 detectors), 29 result(s) found
```

Results

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

Closing Summary

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

Numerous issues were discovered during the initial audit. All of the issues have been fixed by the Qtoken Team.

Disclaimer

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

Audit Report
February, 2022

For

QToken

QuillAudits

Canada, India, Singapore, United Kingdom

audits.quillhash.com

audits@quillhash.com