QuillAudits

Audit Report
August, 2021

Contents

Scope of Audit

The scope of this audit was to analyze and document the Chronic 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 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	0	0
Acknowledged	0	0	4	4
Closed	0	0	0	0

Introduction

During the period of **August 16, 2021 to August 19, 2021** - QuillAudits Team performed a security audit for the Chronic Token smart contract.

The code for the audit was taken from following the official link: https://github.com/ChronicToken/ChronicToken/blob/main/chronicToken.sol

Note	Date	Commit Hash
Version 1	16/08/21	bdc08c623c9b3db912b94a2f3271afd9ad1243ee

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 the contract without 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
   Remove these lock/unlock functions as this seems not to serve a great purpose OR always renounce ownership first before calling the lock function.

   Status: Acknowledged by the Auditee

2. Infinite loop

Line	Code

847	function includeInReward(address account) external onlyOwner() {
	require(_isExcluded[account], "Account is already excluded");
	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;
	}
	}
	}

942	function _getCurrentSupply() private view returns(uint256, uint256) {
	uint256 rSupply = _rTotal;
	uint256 tSupply = _tTotal;
	for (uint256 i = 0; i < _excluded.length; i++) {
	if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return (_rTotal, _tTotal);
	rSupply = rSupply.sub(_rOwned[_excluded[i]]);
	tSupply = tSupply.sub(_tOwned[_excluded[i]]);
	}
	if (rSupply < _rTotal.div(_tTotal)) return (_rTotal,_tTotal);
	return (rSupply, tSupply);
	}

Description

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

Remediation

_exclude length should be limited.

Status: Acknowledged by the Auditee

3. Function input parameters lack of check

Description

Variable validation is not performed in below functions : deliver , setMaxTxPercent , setTaxFeePercent, setBurnFeePercent , setLiquidityFeePercent , _burn.

Remediation
There should be some validations to check the variable is not empty or greater than 0.

Status: Acknowledged by the Auditee

4. Centralized risk in addLiquidity

Line	Code

1125	function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {
	// approve token transfer to cover all possible scenarios
	_approve(address(this), address(uniswapV2Router),tokenAmount);
	// add the liquidity
	uniswapV2Router.addLiquidityETH{value: ethAmount}(
	address(this),
	tokenAmount,
	0, // slippage is unavoidable
	0, // slippage is unavoidable
	owner(),
	block.timestamp
	);
	}

Description
In addLiquidityETH function, owner gets CHT 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 the other hand, the owner can accept this risk and handle the private key very securely.

Status: Acknowledged by the Auditee

Informational

5. Use the latest solidity version

pragma solidity ^0.6.12;

Description
Using the latest solidity will prevent any compiler-level bugs.

Remediation
Please use 0.8.7, which is the latest version.

Status: Acknowledged by the Auditee

6. Make variables constant

Line	Code
697	string private _name = "Chronic Token"; string private _symbol = "CHT"; uint8 private _decimals = 18;
717	uint256 private numTokensSellToAddToLiquidity = 2100000000 * 10**18;

Description
Few variable Values defined will be unchanged. So, please make them constant. It will save some gas.

Remediation
Add Constant keyword for these variables.

Status: Acknowledged by the Auditee

7. Spelling mistake

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

Description
Typing error in comments.

Remediation
Correct the spellings in the comment.

Status: Acknowledged by the Auditee

8. Critical operation lacks event log

Description
Missing event log for : deliver , removeAllFee , excludeFromFee , includeInFee , setTaxFeePercent , setBurnFeePercent , setLiquidityFeePercent , setMaxTxPercent.

Remediation
Please write an event log for listed events.

Status: Acknowledged by the Auditee

Functional test

Function Names	Testing results
constructor	Passed
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	Passed
includeInReward	Infinite loop
_transferBothExcluded	Passed

excludeFromFee	Critical operation lacks event log
includeInFee	Critical operation lacks event log
setTaxFeePercent	Function input parameters lack of check
setBurnFeePercent	Function input parameters lack of check
setLiquidityFeePercent	Function input parameters lack of check
setMaxTxPercent	Function input parameters lack of check
setMaxTxAmount	Passed
setSwapAndLiquifyEnabled	Passed
receive	Spelling mistake
_reflectFee	Passed
_getValues	Passed
_getTValues	Passed
_getRValues	Passed
_getRate	Passed
_getCurrentSupply	Infinite loop
_takeLiquidity	Passed
_takeBurn	Passed
_burn	Function input parameters lack of check
calculateTaxFee	Passed
calculateLiquidityFee	Passed
calculateBurnFee	Passed
removeAllFee	Critical operation lacks event log

restoreAllFee	Passed
isExcludedFromFee	Passed
_approve	Passed
_transfer	Passed
swapAndLiquify	Passed
swapTokensForEth	Passed
addLiquidity	Centralized risk in addLiquidity
_tokenTransfer	Passed
_transferStandard	Passed
_transferToExcluded	Passed
_transferFromExcluded	Passed
owner	Passed
onlyOwner	Passed
renounceOwnership	Possible to gain ownership
transferOwnership	Passed
geUnlockTime	Passed
lock	Possible to gain ownership
unlock	Possible to gain ownership
_msgSender	Passed
_msgData	Passed

Automated Testing

Slither

```
INFO:Detectors:
Reentrancy in ChronicToken._transfer(address,address,uint256) (chronicToken.sol#1038-1082):
        External calls:
        - swapAndLiquify(contractTokenBalance) (chronicToken.sol#1069)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (chronicToken.
sol#1130-1137)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (c
hronicToken.sol#1116-1122)
        External calls sending eth:
        - swapAndLiquify(contractTokenBalance) (chronicToken.sol#1069)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (chronicToken.
sol#1130-1137)
        State variables written after the call(s):
        - _tokenTransfer(from,to,amount,takeFee) (chronicToken.sol#1081)
                - _rOwned[address(0x0)] = _rOwned[address(0x0)].add(rBurn) (chronicToken.sol#965)
                - _rOwned[address(this)] = _rOwned[address(this)].add(rLiquidity) (chronicToken.sol#957)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (chronicToken.sol#1173)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (chronicToken.sol#1163)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (chronicToken.sol#1164)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (chronicToken.sol#1185)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (chronicToken.sol#862)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (chronicToken.sol#1175)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (chronicToken.sol#1186)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (chronicToken.sol#864)
        - _tokenTransfer(from,to,amount,takeFee) (chronicToken.sol#1081)
                - _rTotal = _rTotal.sub(rFee) (chronicToken.sol#910)
        - _tokenTransfer(from,to,amount,takeFee) (chronicToken.sol#1081)
                - _tOwned[address(0x0)] = _tOwned[address(0x0)].add(tBurn) (chronicToken.sol#967)
                - _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (chronicToken.sol#959)
                - _tOwned[sender] = _tOwned[sender].sub(tAmount) (chronicToken.sol#861)
                - _tOwned[sender] = _tOwned[sender].sub(tAmount) (chronicToken.sol#1184)
                - _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (chronicToken.sol#1174)
                - _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (chronicToken.sol#863)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities
INFO:Detectors:
ChronicToken.addLiquidity(uint256,uint256) (chronicToken.sol#1125-1138) ignores return value by uniswapV2Router.addLiquidityETH{value: et
hAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (chronicToken.sol#1130-1137)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
ChronicToken.allowance(address,address).owner (chronicToken.sol#778) shadows:
        - Ownable.owner() (chronicToken.sol#422-424) (function)
ChronicToken._approve(address,address,uint256).owner (chronicToken.sol#1030) shadows:
        - Ownable.owner() (chronicToken.sol#422-424) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
Reentrancy in ChronicToken._transfer(address,address,uint256) (chronicToken.sol#1038-1082):
        External calls:
        - swapAndLiquify(contractTokenBalance) (chronicToken.sol#1069)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (chronicToken.
sol#1130-1137)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (c
hronicToken.sol#1116-1122)
        External calls sending eth:
        - swapAndLiquify(contractTokenBalance) (chronicToken.sol#1069)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (chronicToken.
sol#1130-1137)
        State variables written after the call(s):
        - _tokenTransfer(from,to,amount,takeFee) (chronicToken.sol#1081)
                - _burnFee = _previousBurnFee (chronicToken.sol#1023)
                - _burnFee = 0 (chronicToken.sol#1017)
        - _tokenTransfer(from,to,amount,takeFee) (chronicToken.sol#1081)
                - _liquidityFee = _previousLiquidityFee (chronicToken.sol#1022)
                - _liquidityFee = 0 (chronicToken.sol#1016)
        - _tokenTransfer(from,to,amount,takeFee) (chronicToken.sol#1081)
                - _liquidityFee = 0 (chronicToken.sol#1016)
        - _tokenTransfer(from,to,amount,takeFee) (chronicToken.sol#1081)
                - _previousBurnFee = _burnFee (chronicToken.sol#1013)
        - _tokenTransfer(from,to,amount,takeFee) (chronicToken.sol#1081)
                - _previousLiquidityFee = _liquidityFee (chronicToken.sol#1012)
        - _tokenTransfer(from,to,amount,takeFee) (chronicToken.sol#1081)
                - _previousTaxFee = _taxFee (chronicToken.sol#1011)
```

```
                - _liquidityFee = 0 (chronicToken.sol#1016)
        - _tokenTransfer(from,to,amount,takeFee) (chronicToken.sol#1081)
                - _previousBurnFee = _burnFee (chronicToken.sol#1013)
        - _tokenTransfer(from,to,amount,takeFee) (chronicToken.sol#1081)
                - _previousLiquidityFee = _liquidityFee (chronicToken.sol#1012)
        - _tokenTransfer(from,to,amount,takeFee) (chronicToken.sol#1081)
                - _previousTaxFee = _taxFee (chronicToken.sol#1011)
        - _tokenTransfer(from,to,amount,takeFee) (chronicToken.sol#1081)
                - _tFeeTotal = _tFeeTotal.add(tFee) (chronicToken.sol#911)
        - _tokenTransfer(from,to,amount,takeFee) (chronicToken.sol#1081)
                - _taxFee = _previousTaxFee (chronicToken.sol#1021)
                - _taxFee = 0 (chronicToken.sol#1015)
Reentrancy in ChronicToken.constructor() (chronicToken.sol#733-750):
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (chronicToken.s
ol#739-740)
        State variables written after the call(s):
        - _isExcludedFromFee[owner()] = true (chronicToken.sol#746)
        - _isExcludedFromFee[address(this)] = true (chronicToken.sol#747)
        - uniswapV2Router = _uniswapV2Router (chronicToken.sol#743)
Reentrancy in ChronicToken.swapAndLiquify(uint256) (chronicToken.sol#1084-1105):
        External calls:
        - swapTokensForEth(half) (chronicToken.sol#1096)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (c
hronicToken.sol#1116-1122)
        - addLiquidity(otherHalf,newBalance) (chronicToken.sol#1102)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (chronicToken.
sol#1130-1137)
        External calls sending eth:
        - addLiquidity(otherHalf,newBalance) (chronicToken.sol#1102)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (chronicToken.
sol#1130-1137)
        State variables written after the call(s):
        - addLiquidity(otherHalf,newBalance) (chronicToken.sol#1102)
                - _allowances[owner][spender] = amount (chronicToken.sol#1034)
Reentrancy in ChronicToken.transferFrom(address,address,uint256) (chronicToken.sol#787-791):
        External calls:
        - _transfer(sender,recipient,amount) (chronicToken.sol#788)
```

```
        External calls:
        - _transfer(sender,recipient,amount) (chronicToken.sol#788)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (chronicToken.
sol#1130-1137)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (c
hronicToken.sol#1116-1122)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (chronicToken.sol#788)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (chronicToken.
sol#1130-1137)
        State variables written after the call(s):
        - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (chronicTo
ken.sol#789)
                - _allowances[owner][spender] = amount (chronicToken.sol#1034)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in ChronicToken._transfer(address,address,uint256) (chronicToken.sol#1038-1082):
        External calls:
        - swapAndLiquify(contractTokenBalance) (chronicToken.sol#1069)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (chronicToken.
sol#1130-1137)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (c
hronicToken.sol#1116-1122)
        External calls sending eth:
        - swapAndLiquify(contractTokenBalance) (chronicToken.sol#1069)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (chronicToken.
sol#1130-1137)
        Event emitted after the call(s):
        - Transfer(sender,recipient,tTransferAmount) (chronicToken.sol#1168)
                - _tokenTransfer(from,to,amount,takeFee) (chronicToken.sol#1081)
        - Transfer(sender,recipient,tTransferAmount) (chronicToken.sol#1190)
                - _tokenTransfer(from,to,amount,takeFee) (chronicToken.sol#1081)
        - Transfer(sender,recipient,tTransferAmount) (chronicToken.sol#1179)
                - _tokenTransfer(from,to,amount,takeFee) (chronicToken.sol#1081)
        - Transfer(sender,recipient,tTransferAmount) (chronicToken.sol#868)
                - _tokenTransfer(from,to,amount,takeFee) (chronicToken.sol#1081)
Reentrancy in ChronicToken.constructor() (chronicToken.sol#733-750):
        External calls:
```

```
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (chronicToken.s
ol#739-740)
        Event emitted after the call(s):
        - Transfer(address(0),_msgSender(),_tTotal) (chronicToken.sol#749)
Reentrancy in ChronicToken.swapAndLiquify(uint256) (chronicToken.sol#1084-1105):
        External calls:
        - swapTokensForEth(half) (chronicToken.sol#1096)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (c
hronicToken.sol#1116-1122)
        - addLiquidity(otherHalf,newBalance) (chronicToken.sol#1102)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (chronicToken.
sol#1130-1137)
        External calls sending eth:
        - addLiquidity(otherHalf,newBalance) (chronicToken.sol#1102)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (chronicToken.
sol#1130-1137)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (chronicToken.sol#1035)
                - addLiquidity(otherHalf,newBalance) (chronicToken.sol#1102)
        - SwapAndLiquify(half,newBalance,otherHalf) (chronicToken.sol#1104)
Reentrancy in ChronicToken.transferFrom(address,address,uint256) (chronicToken.sol#787-791):
        External calls:
        - _transfer(sender,recipient,amount) (chronicToken.sol#788)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (chronicToken.
sol#1130-1137)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (c
hronicToken.sol#1116-1122)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (chronicToken.sol#788)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (chronicToken.
sol#1130-1137)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (chronicToken.sol#1035)
                - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (c
hronicToken.sol#789)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
```

```
hronicToken.sol#789)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
Ownable.unlock() (chronicToken.sol#469-474) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(now > _lockTime,Contract is locked until 7 days) (chronicToken.sol#471)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Address.isContract(address) (chronicToken.sol#274-283) uses assembly
        - INLINE ASM (chronicToken.sol#281)
Address._functionCallWithValue(address,bytes,uint256,string) (chronicToken.sol#367-388) uses assembly
        - INLINE ASM (chronicToken.sol#380-383)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address._functionCallWithValue(address,bytes,uint256,string) (chronicToken.sol#367-388) is never used and should be removed
Address.functionCall(address,bytes) (chronicToken.sol#327-329) is never used and should be removed
Address.functionCall(address,bytes,string) (chronicToken.sol#337-339) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (chronicToken.sol#352-354) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (chronicToken.sol#362-365) is never used and should be removed
Address.isContract(address) (chronicToken.sol#274-283) is never used and should be removed
Address.sendValue(address,uint256) (chronicToken.sol#301-307) is never used and should be removed
Context._msgData() (chronicToken.sol#246-249) is never used and should be removed
SafeMath.mod(uint256,uint256) (chronicToken.sol#219-221) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (chronicToken.sol#235-238) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
ChronicToken._rTotal (chronicToken.sol#694) is set pre-construction with a non-constant function or state variable:
        - (MAX - (MAX % _tTotal))
ChronicToken._previousTaxFee (chronicToken.sol#702) is set pre-construction with a non-constant function or state variable:
        - _taxFee
ChronicToken._previousLiquidityFee (chronicToken.sol#705) is set pre-construction with a non-constant function or state variable:
        - _liquidityFee
ChronicToken._previousBurnFee (chronicToken.sol#708) is set pre-construction with a non-constant function or state variable:
        - _burnFee
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#function-initializing-state-variables
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (chronicToken.sol#301-307):
        - (success) = recipient.call{value: amount}() (chronicToken.sol#305)
```

```
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (chronicToken.sol#367-388):
        - (success,returndata) = target.call{value: weiValue}(data) (chronicToken.sol#371)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (chronicToken.sol#508) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (chronicToken.sol#509) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (chronicToken.sol#526) is not in mixedCase
Function IUniswapV2Router01.WETH() (chronicToken.sol#546) is not in mixedCase
Parameter ChronicToken.setSwapAndLiquifyEnabled(bool)._enabled (chronicToken.sol#901) is not in mixedCase
Function ChronicToken._burn(address,uint256) (chronicToken.sol#970-987) is not in mixedCase
Parameter ChronicToken.calculateTaxFee(uint256)._amount (chronicToken.sol#989) is not in mixedCase
```

```
Parameter ChronicToken.setSwapAndLiquifyEnabled(bool)._enabled (chronicToken.sol#901) is not in mixedCase
Function ChronicToken._burn(address,uint256) (chronicToken.sol#970-987) is not in mixedCase
Parameter ChronicToken.calculateTaxFee(uint256)._amount (chronicToken.sol#989) is not in mixedCase
Parameter ChronicToken.calculateLiquidityFee(uint256)._amount (chronicToken.sol#995) is not in mixedCase
Parameter ChronicToken.calculateBurnFee(uint256)._amount (chronicToken.sol#1001) is not in mixedCase
Variable ChronicToken._taxFee (chronicToken.sol#701) is not in mixedCase
Variable ChronicToken._liquidityFee (chronicToken.sol#704) is not in mixedCase
Variable ChronicToken._burnFee (chronicToken.sol#707) is not in mixedCase
Variable ChronicToken._maxTxAmount (chronicToken.sol#716) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (chronicToken.sol#247)" inContext (chronicToken.sol#241-250)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (chronicToken.so
l#551) is too similar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired
(chronicToken.sol#552)
Variable ChronicToken._transferBothExcluded(address,address,uint256).rTransferAmount (chronicToken.sol#860) is too similar to ChronicToke
n._transferBothExcluded(address,address,uint256).tTransferAmount (chronicToken.sol#860)
Variable ChronicToken._getValues(uint256).rTransferAmount (chronicToken.sol#916) is too similar to ChronicToken._getTValues(uint256).tTra
nsferAmount (chronicToken.sol#924)
Variable ChronicToken._transferToExcluded(address,address,uint256).rTransferAmount (chronicToken.sol#1172) is too similar to ChronicToken
._transferToExcluded(address,address,uint256).tTransferAmount (chronicToken.sol#1172)
Variable ChronicToken._transferBothExcluded(address,address,uint256).rTransferAmount (chronicToken.sol#860) is too similar to ChronicToke
n._getTValues(uint256).tTransferAmount (chronicToken.sol#924)
Variable ChronicToken._transferToExcluded(address,address,uint256).rTransferAmount (chronicToken.sol#1172) is too similar to ChronicToken
._getValues(uint256).tTransferAmount (chronicToken.sol#915)
Variable ChronicToken._getValues(uint256).rTransferAmount (chronicToken.sol#916) is too similar to ChronicToken._transferToExcluded(addre
ss,address,uint256).tTransferAmount (chronicToken.sol#1172)
Variable ChronicToken._getRValues(uint256,uint256,uint256,uint256,uint256).rTransferAmount (chronicToken.sol#933) is too similar to Chron
icToken._transferToExcluded(address,address,uint256).tTransferAmount (chronicToken.sol#1172)
Variable ChronicToken._transferFromExcluded(address,address,uint256).rTransferAmount (chronicToken.sol#1183) is too similar to ChronicTok
en._getValues(uint256).tTransferAmount (chronicToken.sol#915)
Variable ChronicToken._transferBothExcluded(address,address,uint256).rTransferAmount (chronicToken.sol#860) is too similar to ChronicToke
n._transferFromExcluded(address,address,uint256).tTransferAmount (chronicToken.sol#1183)
Variable ChronicToken._transferFromExcluded(address,address,uint256).rTransferAmount (chronicToken.sol#1183) is too similar to ChronicTok
en._getTValues(uint256).tTransferAmount (chronicToken.sol#924)
Variable ChronicToken._getValues(uint256).rTransferAmount (chronicToken.sol#916) is too similar to ChronicToken._transferStandard(address
```

```
Variable ChronicToken._getValues(uint256).rTransferAmount (chronicToken.sol#916) is too similar to ChronicToken._transferStandard(address
,address,uint256).tTransferAmount (chronicToken.sol#1162)
Variable ChronicToken._getRValues(uint256,uint256,uint256,uint256,uint256).rTransferAmount (chronicToken.sol#933) is too similar to Chron
icToken._transferStandard(address,address,uint256).tTransferAmount (chronicToken.sol#1162)
Variable ChronicToken._transferBothExcluded(address,address,uint256).rTransferAmount (chronicToken.sol#860) is too similar to ChronicToke
n._getValues(uint256).tTransferAmount (chronicToken.sol#915)
Variable ChronicToken._transferFromExcluded(address,address,uint256).rTransferAmount (chronicToken.sol#1183) is too similar to ChronicTok
en._transferToExcluded(address,address,uint256).tTransferAmount (chronicToken.sol#1172)
Variable ChronicToken._transferBothExcluded(address,address,uint256).rTransferAmount (chronicToken.sol#860) is too similar to ChronicToke
n._transferStandard(address,address,uint256).tTransferAmount (chronicToken.sol#1162)
Variable ChronicToken._getValues(uint256).rTransferAmount (chronicToken.sol#916) is too similar to ChronicToken._getValues(uint256).tTran
sferAmount (chronicToken.sol#915)
Variable ChronicToken._getRValues(uint256,uint256,uint256,uint256,uint256).rTransferAmount (chronicToken.sol#933) is too similar to Chron
icToken._getValues(uint256).tTransferAmount (chronicToken.sol#915)
Variable ChronicToken._transferFromExcluded(address,address,uint256).rTransferAmount (chronicToken.sol#1183) is too similar to ChronicTok
en._transferFromExcluded(address,address,uint256).tTransferAmount (chronicToken.sol#1183)
Variable ChronicToken._transferFromExcluded(address,address,uint256).rTransferAmount (chronicToken.sol#1183) is too similar to ChronicTok
en._transferStandard(address,address,uint256).tTransferAmount (chronicToken.sol#1162)
Variable ChronicToken._transferBothExcluded(address,address,uint256).rTransferAmount (chronicToken.sol#860) is too similar to ChronicToke
n._transferToExcluded(address,address,uint256).tTransferAmount (chronicToken.sol#1172)
Variable ChronicToken._transferFromExcluded(address,address,uint256).rTransferAmount (chronicToken.sol#1183) is too similar to ChronicTok
en._transferBothExcluded(address,address,uint256).tTransferAmount (chronicToken.sol#860)
Variable ChronicToken._transferToExcluded(address,address,uint256).rTransferAmount (chronicToken.sol#1172) is too similar to ChronicToken
._transferStandard(address,address,uint256).tTransferAmount (chronicToken.sol#1162)
Variable ChronicToken._transferStandard(address,address,uint256).rTransferAmount (chronicToken.sol#1162) is too similar to ChronicToken._
transferBothExcluded(address,address,uint256).tTransferAmount (chronicToken.sol#860)
Variable ChronicToken._transferToExcluded(address,address,uint256).rTransferAmount (chronicToken.sol#1172) is too similar to ChronicToken
._transferFromExcluded(address,address,uint256).tTransferAmount (chronicToken.sol#1183)
Variable ChronicToken.reflectionFromToken(uint256,bool).rTransferAmount (chronicToken.sol#826) is too similar to ChronicToken._getTValues
(uint256).tTransferAmount (chronicToken.sol#924)
Variable ChronicToken._getRValues(uint256,uint256,uint256,uint256,uint256).rTransferAmount (chronicToken.sol#933) is too similar to Chron
icToken._getTValues(uint256).tTransferAmount (chronicToken.sol#924)
Variable ChronicToken.reflectionFromToken(uint256,bool).rTransferAmount (chronicToken.sol#826) is too similar to ChronicToken._getValues(
uint256).tTransferAmount (chronicToken.sol#915)
Variable ChronicToken._transferStandard(address,address,uint256).rTransferAmount (chronicToken.sol#1162) is too similar to ChronicToken._
transferToExcluded(address,address,uint256).tTransferAmount (chronicToken.sol#1172)
Variable ChronicToken._getValues(uint256).rTransferAmount (chronicToken.sol#916) is too similar to ChronicToken._transferBothExcluded(add
ress,address,uint256).tTransferAmount (chronicToken.sol#860)
```

```
Variable ChronicToken._getRValues(uint256,uint256,uint256,uint256,uint256).rTransferAmount (chronicToken.sol#933) is too similar to Chron
icToken._transferBothExcluded(address,address,uint256).tTransferAmount (chronicToken.sol#860)
Variable ChronicToken.reflectionFromToken(uint256,bool).rTransferAmount (chronicToken.sol#826) is too similar to ChronicToken._transferTo
Excluded(address,address,uint256).tTransferAmount (chronicToken.sol#1172)
Variable ChronicToken._transferToExcluded(address,address,uint256).rTransferAmount (chronicToken.sol#1172) is too similar to ChronicToken
._transferBothExcluded(address,address,uint256).tTransferAmount (chronicToken.sol#860)
Variable ChronicToken._transferToExcluded(address,address,uint256).rTransferAmount (chronicToken.sol#1172) is too similar to ChronicToken
._getTValues(uint256).tTransferAmount (chronicToken.sol#924)
```

```
Variable ChronicToken.reflectionFromToken(uint256,bool).rTransferAmount (chronicToken.sol#826) is too similar to ChronicToken._transferFr
omExcluded(address,address,uint256).tTransferAmount (chronicToken.sol#1183)
Variable ChronicToken._transferStandard(address,address,uint256).rTransferAmount (chronicToken.sol#1162) is too similar to ChronicToken._
transferStandard(address,address,uint256).tTransferAmount (chronicToken.sol#1162)
Variable ChronicToken._transferStandard(address,address,uint256).rTransferAmount (chronicToken.sol#1162) is too similar to ChronicToken._
getTValues(uint256).tTransferAmount (chronicToken.sol#924)
Variable ChronicToken._getValues(uint256).rTransferAmount (chronicToken.sol#916) is too similar to ChronicToken._transferFromExcluded(add
ress,address,uint256).tTransferAmount (chronicToken.sol#1183)
Variable ChronicToken._getRValues(uint256,uint256,uint256,uint256,uint256).rTransferAmount (chronicToken.sol#933) is too similar to Chron
icToken._transferFromExcluded(address,address,uint256).tTransferAmount (chronicToken.sol#1183)
Variable ChronicToken._transferStandard(address,address,uint256).rTransferAmount (chronicToken.sol#1162) is too similar to ChronicToken._
transferFromExcluded(address,address,uint256).tTransferAmount (chronicToken.sol#1183)
Variable ChronicToken.reflectionFromToken(uint256,bool).rTransferAmount (chronicToken.sol#826) is too similar to ChronicToken._transferSt
andard(address,address,uint256).tTransferAmount (chronicToken.sol#1162)
Variable ChronicToken._transferStandard(address,address,uint256).rTransferAmount (chronicToken.sol#1162) is too similar to ChronicToken._
getValues(uint256).tTransferAmount (chronicToken.sol#915)
Variable ChronicToken.reflectionFromToken(uint256,bool).rTransferAmount (chronicToken.sol#826) is too similar to ChronicToken._transferBo
thExcluded(address,address,uint256).tTransferAmount (chronicToken.sol#860)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
ChronicToken.slitherConstructorVariables() (chronicToken.sol#679-1193) uses literals with too many digits:
        - _tTotal = 4200000000 * 10 ** 18 (chronicToken.sol#693)
ChronicToken.slitherConstructorVariables() (chronicToken.sol#679-1193) uses literals with too many digits:
        - _maxTxAmount = 4200000000 * 10 ** 18 (chronicToken.sol#716)
ChronicToken.slitherConstructorVariables() (chronicToken.sol#679-1193) uses literals with too many digits:
        - numTokensSellToAddToLiquidity = 2100000000 * 10 ** 18 (chronicToken.sol#717)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
ChronicToken._decimals (chronicToken.sol#699) should be constant
ChronicToken._name (chronicToken.sol#697) should be constant
ChronicToken._symbol (chronicToken.sol#698) should be constant
ChronicToken.numTokensSellToAddToLiquidity (chronicToken.sol#717) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (chronicToken.sol#441-444)
transferOwnership(address) should be declared external:
```

```
        - Ownable.renounceOwnership() (chronicToken.sol#441-444)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (chronicToken.sol#450-454)
geUnlockTime() should be declared external:
        - Ownable.geUnlockTime() (chronicToken.sol#456-458)
lock(uint256) should be declared external:
        - Ownable.lock(uint256) (chronicToken.sol#461-466)
unlock() should be declared external:
        - Ownable.unlock() (chronicToken.sol#469-474)
name() should be declared external:
        - ChronicToken.name() (chronicToken.sol#752-754)
symbol() should be declared external:
        - ChronicToken.symbol() (chronicToken.sol#756-758)
decimals() should be declared external:
        - ChronicToken.decimals() (chronicToken.sol#760-762)
totalSupply() should be declared external:
        - ChronicToken.totalSupply() (chronicToken.sol#764-766)
transfer(address,uint256) should be declared external:
        - ChronicToken.transfer(address,uint256) (chronicToken.sol#773-776)
allowance(address,address) should be declared external:
        - ChronicToken.allowance(address,address) (chronicToken.sol#778-780)
approve(address,uint256) should be declared external:
        - ChronicToken.approve(address,uint256) (chronicToken.sol#782-785)
transferFrom(address,address,uint256) should be declared external:
        - ChronicToken.transferFrom(address,address,uint256) (chronicToken.sol#787-791)
increaseAllowance(address,uint256) should be declared external:
        - ChronicToken.increaseAllowance(address,uint256) (chronicToken.sol#793-796)
decreaseAllowance(address,uint256) should be declared external:
        - ChronicToken.decreaseAllowance(address,uint256) (chronicToken.sol#798-801)
isExcludedFromReward(address) should be declared external:
        - ChronicToken.isExcludedFromReward(address) (chronicToken.sol#803-805)
totalFees() should be declared external:
        - ChronicToken.totalFees() (chronicToken.sol#807-809)
deliver(uint256) should be declared external:
        - ChronicToken.deliver(uint256) (chronicToken.sol#811-818)
reflectionFromToken(uint256,bool) should be declared external:
        - ChronicToken.reflectionFromToken(uint256,bool) (chronicToken.sol#820-829)
excludeFromReward(address) should be declared external:
```

```
excludeFromReward(address) should be declared external:
        - ChronicToken.excludeFromReward(address) (chronicToken.sol#837-845)
excludeFromFee(address) should be declared external:
        - ChronicToken.excludeFromFee(address) (chronicToken.sol#871-873)
includeInFee(address) should be declared external:
        - ChronicToken.includeInFee(address) (chronicToken.sol#875-877)
setSwapAndLiquifyEnabled(bool) should be declared external:
        - ChronicToken.setSwapAndLiquifyEnabled(bool) (chronicToken.sol#901-904)
_burn(address,uint256) should be declared external:
        - ChronicToken._burn(address,uint256) (chronicToken.sol#970-987)
isExcludedFromFee(address) should be declared external:
        - ChronicToken.isExcludedFromFee(address) (chronicToken.sol#1026-1028)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:chronicToken.sol analyzed (10 contracts with 75 detectors), 120 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

SOLIDITY STATIC ANALYSIS

chronicToken.sol

Security

Transaction origin:

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

Check-effects-interaction:

INTERNAL ERROR in module Check-effects-interaction: can't convert undefined to object
Pos: not available

Inline assembly:

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

Block timestamp:

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

Low level calls:

INTERNAL ERROR in module Low level calls: can't convert undefined to object
Pos: not available

Selfdestruct:

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

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

Similar variable names:

INTERNAL ERROR in module Similar variable names: can't convert undefined to object
Pos: not available

No return:

INTERNAL ERROR in module No return: can't convert undefined to object
Pos: not available

Guard conditions:

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

String length:

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

SOLHINT LINTER

```
 chronicToken.sol:11:1: Error: Compiler version ^0.6.12 does not satisfy the r semver requirement

 chronicToken.sol:464:21: Error: Avoid to make time-based decisions in your business logic

 chronicToken.sol:471:17: Error: Avoid to make time-based decisions in your business logic

 chronicToken.sol:508:5: Error: Function name must be in mixedCase

 chronicToken.sol:509:5: Error: Function name must be in mixedCase

 chronicToken.sol:526:5: Error: Function name must be in mixedCase

 chronicToken.sol:546:5: Error: Function name must be in mixedCase

 chronicToken.sol:679:1: Error: Contract has 22 states declarations but allowed no more than 15

 chronicToken.sol:713:5: Error: Explicitly mark visibility of state

 chronicToken.sol:907:32: Error: Code contains empty blocks

 chronicToken.sol:1121:13: Error: Avoid to make time-based decisions in your business logic

 chronicToken.sol:1136:13: Error: Avoid to make time-based decisions in your business logic
```

Disclaimer

Quillhash audit is not a security warranty, investment advice, or an endorsement of the Chronic Token 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 Chronic Token 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.

All of the concerns addressed above have been acknowledged by Chronic Token Team.

QuillAudits

Canada, India, Singapore and United Kingdom

audits.quillhash.com

audits@quillhash.com