



# Smart contracts security assessment

Final report

[Tariff: Standard](#)

## Dragon Crown Router

February 2024



[0xguard.com](https://0xguard.com)



[hello@0xguard.com](mailto:hello@0xguard.com)

## Contents

1. Introduction	3
2. Contracts checked	3
3. Procedure	3
4. Known vulnerabilities checked	4
5. Classification of issue severity	5
6. Issues	5
7. Conclusion	7
8. Disclaimer	8
9. Slither output	9

## Introduction

The report has been prepared for **Dragon Crown Router**.

The project is a UniswapV2 Router fork with minor additions. The only modified function is `_swapSupportingFeeOnTransferTokens` that includes an alternative calculation of the pair's balance based on external calls that are out of the scope of this audit.

**The SHA-1 hashes of audited files are:**

Router.sol `84e25e34114180b0eb501858c77eca3c95299a0b`

**A recheck was done for the following files with SHA-1 hashes:**

Router.sol `89d28556779e2b4c6a45585995e6c8f9fbf80f6e`

Name	Dragon Crown Router
Audit date	2024-02-07 - 2024-02-08
Language	Solidity
Platform	Arbitrum Network

## Contracts checked

Name	Address
Router.sol	0x3C04e848bd3a93E05222803D11eFBf04971aa224

## Procedure

We perform our audit according to the following procedure:

**Automated analysis**

- Scanning the project's smart contracts with several publicly available automated Solidity analysis tools
- Manual verification (reject or confirm) all the issues found by the tools

### Manual audit

- Manually analyze smart contracts for security vulnerabilities
- Smart contracts' logic check

## Known vulnerabilities checked

Title	Check result
<u>Unencrypted Private Data On-Chain</u>	passed
<u>Code With No Effects</u>	passed
<u>Message call with hardcoded gas amount</u>	passed
<u>Typographical Error</u>	passed
<u>DoS With Block Gas Limit</u>	passed
<u>Presence of unused variables</u>	passed
<u>Incorrect Inheritance Order</u>	passed
<u>Requirement Violation</u>	passed
<u>Weak Sources of Randomness from Chain Attributes</u>	passed
<u>Shadowing State Variables</u>	passed
<u>Incorrect Constructor Name</u>	passed
<u>Block values as a proxy for time</u>	passed
<u>Authorization through tx.origin</u>	passed
<u>DoS with Failed Call</u>	passed
<u>Delegatecall to Untrusted Callee</u>	passed

<u>Use of Deprecated Solidity Functions</u>	passed
<u>Assert Violation</u>	passed
<u>State Variable Default Visibility</u>	passed
<u>Reentrancy</u>	passed
<u>Unprotected SELFDESTRUCT Instruction</u>	passed
<u>Unprotected Ether Withdrawal</u>	passed
<u>Unchecked Call Return Value</u>	passed
<u>Floating Pragma</u>	passed
<u>Outdated Compiler Version</u>	passed
<u>Integer Overflow and Underflow</u>	passed
<u>Function Default Visibility</u>	passed

## Classification of issue severity

<b>High severity</b>	High severity issues can cause a significant or full loss of funds, change of contract ownership, major interference with contract logic. Such issues require immediate attention.
<b>Medium severity</b>	Medium severity issues do not pose an immediate risk, but can be detrimental to the client's reputation if exploited. Medium severity issues may lead to a contract failure and can be fixed by modifying the contract state or redeployment. Such issues require attention.
<b>Low severity</b>	Low severity issues do not cause significant destruction to the contract's functionality. Such issues are recommended to be taken into consideration.

## Issues

**High severity issues**

No issues were found

**Medium severity issues**

No issues were found

**Low severity issues**

No issues were found

## Conclusion

Dragon Crown Router Router.sol contract was audited. No severity issues were found.

The audit did not identify any security issues within the Router contract itself. However, the Router contract interacts with a custom Pair contract, which is out of scope. To fully ensure correct functionality, the Pair contract should also be audited.

## Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to the Company in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes without 0xGuard prior written consent.

This report is not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team or project that contracts 0xGuard to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.



## Slither output

INFO:Detectors:

DCONRouter.removeLiquidity(address,address,uint256,uint256,uint256,address,uint256) (Router (1).sol#750-774) ignores return value by

IDCONPair(pair).transferFrom(msg.sender,pair,liquidity) (Router (1).sol#766)

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-transfer>

INFO:Detectors:

DCONRouter.\_swapSupportingFeeOnTransferTokens(address[],address).i (Router (1).sol#1136) is a local variable never initialized

DCONRouter.\_swap(uint256[],address[],address).i (Router (1).sol#939) is a local variable never initialized

DCONLibrary.getAmountsOut(address,uint256,address[]).i (Router (1).sol#538) is a local variable never initialized

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables>

INFO:Detectors:

DCONRouter.\_addLiquidity(address,address,uint256,uint256,uint256,uint256) (Router (1).sol#639-684) ignores return value by

IDCONFactory(factory).createPair(tokenA,tokenB) (Router (1).sol#649)

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return>

INFO:Detectors:

DCONRouter.constructor(address,address).\_factory (Router (1).sol#629) lacks a zero-check on :

- factory = \_factory (Router (1).sol#630)

DCONRouter.constructor(address,address).\_WETH (Router (1).sol#629) lacks a zero-check on :

- WETH = \_WETH (Router (1).sol#631)

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation>

INFO:Detectors:

Different versions of Solidity are used:

- Version used: ['=0.6.6', '>=0.5.0', '>=0.6.0', '>=0.6.2']
- =0.6.6 (Router (1).sol#297)
- =0.6.6 (Router (1).sol#616)
- >=0.5.0 (Router (1).sol#260)
- >=0.5.0 (Router (1).sol#317)
- >=0.5.0 (Router (1).sol#428)

- >=0.5.0 (Router (1).sol#570)
- >=0.5.0 (Router (1).sol#604)
- >=0.6.0 (Router (1).sol#2)
- >=0.6.2 (Router (1).sol#56)
- >=0.6.2 (Router (1).sol#209)

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used>

INFO:Detectors:

TransferHelper.safeApprove(address,address,uint256) (Router (1).sol#10-19) is never used and should be removed

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code>

INFO:Detectors:

Pragma version>=0.6.0 (Router (1).sol#2) allows old versions

Pragma version>=0.6.2 (Router (1).sol#56) allows old versions

Pragma version>=0.6.2 (Router (1).sol#209) allows old versions

Pragma version>=0.5.0 (Router (1).sol#260) allows old versions

Pragma version=0.6.6 (Router (1).sol#297) allows old versions

Pragma version>=0.5.0 (Router (1).sol#317) allows old versions

Pragma version>=0.5.0 (Router (1).sol#428) allows old versions

Pragma version>=0.5.0 (Router (1).sol#570) allows old versions

Pragma version>=0.5.0 (Router (1).sol#604) allows old versions

Pragma version=0.6.6 (Router (1).sol#616) allows old versions

solc-0.6.6 is not recommended for deployment

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity>

INFO:Detectors:

Low level call in TransferHelper.safeApprove(address,address,uint256) (Router (1).sol#10-19):

- (success,data) = token.call(abi.encodeWithSelector(0x095ea7b3,to,value)) (Router (1).sol#12-14)

Low level call in TransferHelper.safeTransfer(address,address,uint256) (Router (1).sol#21-30):

- (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,value)) (Router (1).sol#23-25)

Low level call in TransferHelper.safeTransferFrom(address,address,address,uint256) (Router (1).sol#32-46):

- (success,data) = token.call(abi.encodeWithSelector(0x23b872dd,from,to,value)) (Router (1).sol#39-41)

Low level call in TransferHelper.safeTransferETH(address,uint256) (Router (1).sol#48-51):

- (success) = to.call{value: value}(new bytes(0)) (Router (1).sol#49)

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls>

INFO:Detectors:

Function IDCONRouter01.WETH() (Router (1).sol#61) is not in mixedCase

Function IDCONFactory.INIT\_CODE\_PAIR\_HASH() (Router (1).sol#292) is not in mixedCase

Function IDCONPair.DOMAIN\_SEPARATOR() (Router (1).sol#348) is not in mixedCase

Function IDCONPair.PERMIT\_TYPEHASH() (Router (1).sol#350) is not in mixedCase

Function IDCONPair.MINIMUM\_LIQUIDITY() (Router (1).sol#381) is not in mixedCase

Variable DCONRouter.WETH (Router (1).sol#622) is not in mixedCase

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions>

INFO:Detectors:

Variable IDCONRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (Router (1).sol#66) is too similar to IDCONRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (Router (1).sol#67)

Variable DCONRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (Router (1).sol#689) is too similar to DCONRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (Router (1).sol#690)

Variable DCONRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (Router (1).sol#689) is too similar to IDCONRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (Router (1).sol#67)

Variable IDCONRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (Router (1).sol#66) is too similar to DCONRouter.\_addLiquidity(address,address,uint256,uint256,uint256,uint256).amountBDesired (Router (1).sol#643)

Variable DCONRouter.\_addLiquidity(address,address,uint256,uint256,uint256,uint256).amountADesired (Router (1).sol#642) is too similar to DCONRouter.\_addLiquidity(address,address,uint256,uint256,uint256,uint256).amountBDesired (Router (1).sol#643)

Variable DCONRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (Router (1).sol#689) is too similar to DCONRouter.\_addLiquidity(address,address,uint256,uint256,uint256,uint256).amountBDesired (Router (1).sol#643)

Variable DCONRouter.\_addLiquidity(address,address,uint256,uint256,uint256,uint256).amountADesired (Router (1).sol#642) is too similar to DCONRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (Router (1).sol#690)

Variable IDCONRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (Router (1).sol#66) is too similar to DCONRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (Router (1).sol#690)

Variable DCONRouter.\_addLiquidity(address,address,uint256,uint256,uint256,uint256).amount

tADesired (Router (1).sol#642) is too similar to IDCONRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (Router (1).sol#67) Variable DCONRouter.\_addLiquidity(address,address,uint256,uint256,uint256,uint256).amountAOptimal (Router (1).sol#671-675) is too similar to DCONRouter.\_addLiquidity(address,address,uint256,uint256,uint256,uint256).amountBOptimal (Router (1).sol#659-663) Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar>

INFO:Detectors:

getAmountsOut(uint256,address[]) should be declared external:

- DCONRouter.getAmountsOut(uint256,address[]) (Router (1).sol#1285-1290)

Moreover, the following function parameters should change its data location:

path location should be calldata

getAmountsIn(uint256,address[]) should be declared external:

- DCONRouter.getAmountsIn(uint256,address[]) (Router (1).sol#1292-1297)

Moreover, the following function parameters should change its data location:

path location should be calldata

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external>

INFO:Slither:Router (1).sol analyzed (10 contracts with 85 detectors), 42 result(s) found



 Guard