Guard

Smart contracts security assessment

Final report

Tariff: Standard

Dragon Crown Router

February 2024

0xguard.com

hello@0xguard.com

Contents

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

Name	Dragon Crown Router
Audit date	2024-02-07 - 2024-02-08
Language	Solidity
Platform	Binance Smart Chain

Contracts checked

Name	Address
Router.sol	

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
Unencrypted Private Data On-Chain	passed
Code With No Effects	passed
Message call with hardcoded gas amount	passed
Typographical Error	passed
DoS With Block Gas Limit	passed
Presence of unused variables	passed
Incorrect Inheritance Order	passed
Requirement Violation	passed
Weak Sources of Randomness from Chain Attributes	passed
Shadowing State Variables	passed
Incorrect Constructor Name	passed
Block values as a proxy for time	passed
Authorization through tx.origin	passed
DoS with Failed Call	passed
Delegatecall to Untrusted Callee	passed
Use of Deprecated Solidity Functions	passed
Assert Violation	passed
State Variable Default Visibility	passed

Reentrancy	passed
Unprotected SELFDESTRUCT Instruction	passed
Unprotected Ether Withdrawal	passed
Unchecked Call Return Value	passed
Floating Pragma	passed
Outdated Compiler Version	passed
Integer Overflow and Underflow	passed
Function Default Visibility	passed

Classification of issue severity

High severity	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.
Medium severity	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.
Low severity	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:
DCRNRouter.removeLiquidity(address,address,uint256,uint256,uint256,address,uint256)
(contracts/Router.sol#741-765) ignores return value by
IDCRNPair(pair).transferFrom(msg.sender,pair,liquidity) (contracts/Router.sol#757)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-
transfer
INFO:Detectors:
DCRNLibrary.getReserves(address,address,address) (contracts/Router.sol#458-471) ignores
return value by (reserve0,reserve1) =
IDCRNPair(pairFor(factory,tokenA,tokenB)).getReserves() (contracts/Router.sol#465-467)
DCRNRouter._addLiquidity(address,address,uint256,uint256,uint256,uint256) (contracts/
Router.sol#630-675) ignores return value by
IDCRNFactory(factory).createPair(tokenA,tokenB) (contracts/Router.sol#640)
DCRNRouter.removeLiquidity(address,address,uint256,uint256,uint256,address,uint256)
(contracts/Router.sol#741-765) ignores return value by (token0) =
DCRNLibrary.sortTokens(tokenA,tokenB) (contracts/Router.sol#759)
DCRNRouter._swap(uint256[],address[],address) (contracts/Router.sol#925-947) ignores
return value by (token0) = DCRNLibrary.sortTokens(input,output) (contracts/
Router.sol#932)
DCRNRouter._swapSupportingFeeOnTransferTokens(address[],address) (contracts/
Router.sol#1123-1176) ignores return value by (token0) =
DCRNLibrary.sortTokens(input,output) (contracts/Router.sol#1129)
DCRNRouter._swapSupportingFeeOnTransferTokens(address[],address) (contracts/
Router.sol#1123-1176) ignores return value by (reserve0,reserve1) = pair.getReserves()
(contracts/Router.sol#1137)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
DCRNRouter.constructor(address,address)._factory (contracts/Router.sol#620) lacks a
zero-check on :
                - factory = _factory (contracts/Router.sol#621)
DCRNRouter.constructor(address,address)._WETH (contracts/Router.sol#620) lacks a zero-
check on :
                - WETH = _WETH (contracts/Router.sol#622)
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 (contracts/Router.sol#292)
        - =0.6.6 (contracts/Router.sol#607)
        - >=0.5.0 (contracts/Router.sol#255)
        - >=0.5.0 (contracts/Router.sol#312)
        - >=0.5.0 (contracts/Router.sol#419)
        - >=0.5.0 (contracts/Router.sol#561)
        - >=0.5.0 (contracts/Router.sol#595)
        - >=0.6.0 (contracts/Router.sol#1)
        - >=0.6.2 (contracts/Router.sol#51)
        - >=0.6.2 (contracts/Router.sol#204)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-
pragma-directives-are-used
INFO:Detectors:
TransferHelper.safeApprove(address,address,uint256) (contracts/Router.sol#5-14) 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 (contracts/Router.sol#1) allows old versions
Pragma version>=0.6.2 (contracts/Router.sol#51) allows old versions
Pragma version>=0.6.2 (contracts/Router.sol#204) allows old versions
Pragma version>=0.5.0 (contracts/Router.sol#255) allows old versions
Pragma version=0.6.6 (contracts/Router.sol#292) allows old versions
Pragma version>=0.5.0 (contracts/Router.sol#312) allows old versions
Pragma version>=0.5.0 (contracts/Router.sol#419) allows old versions
Pragma version>=0.5.0 (contracts/Router.sol#561) allows old versions
Pragma version>=0.5.0 (contracts/Router.sol#595) allows old versions
Pragma version=0.6.6 (contracts/Router.sol#607) 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) (contracts/
Router.sol#5-14):
        - (success,data) = token.call(abi.encodeWithSelector(0x095ea7b3,to,value))
(contracts/Router.sol#7-9)
Low level call in TransferHelper.safeTransfer(address,address,uint256) (contracts/
Router.sol#16-25):
        - (success,data) = token.call(abi.encodeWithSelector(0xa9059cbb,to,value))
(contracts/Router.sol#18-20)
Low level call in TransferHelper.safeTransferFrom(address,address,address,uint256)
```

```
(contracts/Router.sol#27-41):
        - (success,data) = token.call(abi.encodeWithSelector(0x23b872dd,from,to,value))
(contracts/Router.sol#34-36)
Low level call in TransferHelper.safeTransferETH(address,uint256) (contracts/
Router.sol#43-46):
        - (success) = to.call{value: value}(new bytes(0)) (contracts/Router.sol#44)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-
calls
INFO:Detectors:
Function IDCRNRouter01.WETH() (contracts/Router.sol#56) is not in mixedCase
Function IDCRNFactory.INIT_CODE_PAIR_HASH() (contracts/Router.sol#287) is not in
mixedCase
Function IDCRNPair.DOMAIN_SEPARATOR() (contracts/Router.sol#343) is not in mixedCase
Function IDCRNPair.PERMIT_TYPEHASH() (contracts/Router.sol#345) is not in mixedCase
Function IDCRNPair.MINIMUM_LIQUIDITY() (contracts/Router.sol#376) is not in mixedCase
Variable DCRNRouter.WETH (contracts/Router.sol#613) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-
solidity-naming-conventions
INFO:Detectors:
Variable IDCRNRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,addr
ess,uint256).amountADesired (contracts/Router.sol#61) is too similar to IDCRNRouter01.ad
dLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesir
ed (contracts/Router.sol#62)
Variable DCRNRouter._addLiquidity(address,address,uint256,uint256,uint256,uint256).amoun
tADesired (contracts/Router.sol#633) is too similar to DCRNRouter._addLiquidity(address,
address,uint256,uint256,uint256,uint256).amountBDesired (contracts/Router.sol#634)
Variable DCRNRouter._addLiquidity(address,address,uint256,uint256,uint256,uint256).amoun
tADesired (contracts/Router.sol#633) is too similar to IDCRNRouter01.addLiquidity(addres
s,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/
Router.sol#62)
Variable IDCRNRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,addr
ess,uint256).amountADesired (contracts/Router.sol#61) is too similar to DCRNRouter.addLi
quidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired
(contracts/Router.sol#681)
Variable IDCRNRouter01.addLiquidity(address,address,uint256,uint256,uint256,uint256,addr
ess,uint256).amountADesired (contracts/Router.sol#61) is too similar to DCRNRouter._addL
iquidity(address,address,uint256,uint256,uint256,uint256).amountBDesired (contracts/
Router.sol#634)
Variable DCRNRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,address
,uint256).amountADesired (contracts/Router.sol#680) is too similar to DCRNRouter.addLiqu
idity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired
```

```
(contracts/Router.sol#681)
Variable DCRNRouter._addLiquidity(address,address,uint256,uint256,uint256,uint256).amoun
tADesired (contracts/Router.sol#633) is too similar to DCRNRouter.addLiquidity(address,a
ddress,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/
Router.sol#681)
Variable DCRNRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,address
,uint256).amountADesired (contracts/Router.sol#680) is too similar to IDCRNRouter01.addL
iquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired
 (contracts/Router.sol#62)
Variable DCRNRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,address
,uint256).amountADesired (contracts/Router.sol#680) is too similar to DCRNRouter._addLiq
uidity(address,address,uint256,uint256,uint256,uint256).amountBDesired (contracts/
Router.sol#634)
Variable DCRNRouter._addLiquidity(address,address,uint256,uint256,uint256,uint256).amoun
tAOptimal (contracts/Router.sol#662-666) is too similar to DCRNRouter._addLiquidity(addr
ess,address,uint256,uint256,uint256,uint256).amountBOptimal (contracts/
Router.sol#650-654)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-
too-similar
INFO:Detectors:
getAmountsOut(uint256,address[]) should be declared external:
        - DCRNRouter.getAmountsOut(uint256,address[]) (contracts/Router.sol#1273-1278)
Moreover, the following function parameters should change its data location:
path location should be calldata
getAmountsIn(uint256,address[]) should be declared external:
        - DCRNRouter.getAmountsIn(uint256,address[]) (contracts/Router.sol#1280-1285)
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:. analyzed (10 contracts with 88 detectors), 44 result(s) found
```

0x Guard