QuillAudits

Audit Report
July, 2021

KUCOIN LAUNCHPAD

Contents

Scope of Audit

The scope of this audit was to analyze and document the KUCOIN LAUNCHPAD 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	0	0
Closed	0	1	3	6

Introduction

During the period of **July 19, 2021 to July 20, 2021** - QuillAudits Team performed a security audit for KUCOIN LAUNCHPAD smart contracts.

The code for the audit was taken from following the official link:

Note	Date	Code
Version 1	July	https://bscscan.com/address/0xf74b56ce2e8240c4c91632b8337781304b02d1cc#code
Version 2	July	https://bscscan.com/address/0xda51e0c8adde37781e57f3fb180b8ad1b8ae3d17#code
Version 3	July	https://bscscan.com/token/0x47b8806c2891c4a92b5c590c32cfe1eb617648ef

Issues Found – Code Review / Manual Testing

High severity issues

No issues were found

Medium severity issues

1. **Loops in the Contract are extremely costly**

   Description
   The for loop in the release() function includes state variables .length of a non-memory array, in the condition of the for loops. As a result, these state variables consume a lot more extra gas for every iteration of the for loop.

   Remediation
   We recommend using a local variable instead of a state variable .length in a loop.

   Status: Fixed
   This issue was fixed in Version 2.

Low level severity issues

2. **Missing error message for require functions.**

   Description
   The following require functions are missing the message error:
   L61: require (balances[msg.sender]>=_amount && _amount>0 && balances[_to]+_amount>balances[_to]);

   L70: require (balances[_from]>=_amount && allowed[_from][msg.sender]>=_amount && _amount>0 && balances[_to]+_amount>balances[_to]);

   Remediation
   We recommend adding the message error for each require function listed above.

Status: Fixed
This issue was fixed in Version 2.

3. msg.sender is not checked

Description
The msg.sender should be validated if it exists in the lockedAccount before further processing.

Remediation
We recommend adding the require function to check if the msg.sender has been added to lockedAccount. This helps to avoid the potential for erroneous values to result in unexpected behaviors or wasted gas.

Status: Fixed
This issue was fixed in Version 3.

4. Missing Check for Reentrancy Attack

Description
The following functions do not perform zero address validation:

- transfer()
- transferFrom()
- approve()
- timelock()
- mint()
- burn()

Remediation
Consider implementing require statements where appropriate to validate all user-controlled input, including constructor, to avoid the potential for erroneous values to result in unexpected behaviors or wasted gas.

Status: Fixed
This issue was fixed in Version 3.

Informational

5. State Variable Default Visibility

Line	Code
52-53	mapping (address=>uint256) balances; mapping (address=>mapping (address=>uint256)) allowed;
129	mapping(address => LockedAccounts) lock;

Description
The Visibility of the above-mentioned variable is not defined. Labeling the visibility explicitly makes it easier to catch incorrect assumptions about who can access the variable.

The default is internal for state variables, but it should be made explicit.

Remediation
We recommend adding the visibility for these variables. Variables can be specified as being public, internal or private. Explicitly define visibility for all state variables.

Status: Fixed
This issue was fixed in Version 3.

6. 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:

- timelock()
- release()
- lockedAccountDetails()

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

Status: Fixed
This issue was fixed in Version 3.

7. Not using delete to zero values

Description

In the release() function, the following variables are manually replaced with Zero:

- loc.amount = 0;
- loc.time = 0;
- loc.lockedAt = 0;

Remediation

To simplify the code and clarify intent, consider using delete instead.

Status: Fixed

This issue was fixed in Version 3.

8. Missing docstrings

Description

It is extremely difficult to locate any contracts or functions, as they lack documentation. One consequence of this is that reviewers' understanding of the code's intention is impeded, which is significant because it is necessary to accurately determine both security and correctness.

They are additionally more readable and easier to maintain when wrapped in docstrings. The functions should be documented so that users can understand the purpose or intention of each function, as well as the situations in which it may fail, who is allowed to call it, what values it returns, and what events it emits.

Remediation

Consider thoroughly documenting all functions (and their parameters) that are part of the contracts' public API. Functions implementing sensitive functionality, even if those are not public, should be clearly documented as well. When writing docstrings, consider following the Ethereum Natural Specification Format (NatSpec).

Status: Fixed

This issue was fixed in Version 3.

9. Use double quotes for string literals

Line	Code
191	require(msg.sender == owner, 'only admin');
196	require(msg.sender == owner, 'only admin');

Description
Single quote found in the above string variables. Whilst, the double quotes are being utilized for other string literals.

Remediation
We recommend using double quotes for string literals.

Status: Fixed
This issue was fixed in Version 3.

10. Conformance to Solidity naming conventions

Line	Code
131	timelock()

Description
Functions other than constructors should use mixedCase. Examples: getBalance, transfer, verifyOwner, addMember, changeOwner.

Remediation
Follow the Solidity naming convention. We recommend changing the function name timelock to timeLock

Status: Fixed
This issue was fixed in Version 3.

Functional test

Function Names	Testing results
approve	Passed
burn	Passed
changeFreezeStatus	Passed
changeOwnerShip	Passed
excludeFromReward	Passed
mint	Passed
release	Passed
timelock	Passed
transfer	Passed
transferFrom	Passed

Automated Testing

Slither

```
INFO:Detectors:
TimeLock.release() (kucoin.sol#141-155) ignores return value by
BEP(address(this)).transfer(msg.sender,amount) (kucoin.sol#149)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-
transfer
INFO:Detectors:
Owned.changeOwnership(address) (kucoin.sol#15-17) should emit an event for:
    - owner = _newOwner (kucoin.sol#16)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-
access-control
INFO:Detectors:
Owned.changeOwnership(address)._newOwner (kucoin.sol#15) lacks a zero-check on :
        - owner = _newOwner (kucoin.sol#16)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-
address-validation
INFO:Detectors:
TimeLock.release() (kucoin.sol#141-155) uses timestamp for comparisons
    Dangerous comparisons:
    - require(bool,string)(block.timestamp >= (loc.time + loc.lockedAt),TimeLock: Release time
not reached) (kucoin.sol#147)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Freeze.freezeCheck() (kucoin.sol#22-25) compares to a boolean constant:
    -require(bool,string)(isFreeze == false,Contract have frozen) (kucoin.sol#23)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#boolean-equality
INFO:Detectors:
Pragma version0.8.4 (kucoin.sol#6) necessitates a version too recent to be trusted. Consider
deploying with 0.6.12/0.7.6
solc-0.8.4 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity
INFO:Detectors:
Parameter Owned.changeOwnership(address)._newOwner (kucoin.sol#15) is not in mixedCase
Parameter BEP20.balanceOf(address)._owner (kucoin.sol#57) is not in mixedCase
Parameter BEP20.transfer(address,uint256)._to (kucoin.sol#59) is not in mixedCase
Parameter BEP20.transfer(address,uint256)._amount (kucoin.sol#59) is not in mixedCase
Parameter BEP20.transferFrom(address,address,uint256)._from (kucoin.sol#68) is not in
mixedCase
Parameter BEP20.transferFrom(address,address,uint256)._to (kucoin.sol#68) is not in
mixedCase
```

```
Parameter BEP20.transferFrom(address,address,uint256)._amount (kucoin.sol#68) is not in
mixedCase
Parameter BEP20.approve(address,uint256)._spender (kucoin.sol#78) is not in mixedCase
Parameter BEP20.approve(address,uint256)._amount (kucoin.sol#78) is not in mixedCase
Parameter BEP20.allowance(address,address)._owner (kucoin.sol#84) is not in mixedCase
Parameter BEP20.allowance(address,address)._spender (kucoin.sol#84) is not in mixedCase
Parameter TimeLock.timelock(address,uint256,uint256)._lockAccount (kucoin.sol#131) is not in
mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-
solidity-naming-conventions
INFO:Detectors:
KuCoinLaunchPad.constructor() (kucoin.sol#180-188) uses literals with too many digits:
    - totalSupply = 350000000 * 10 ** 18 (kucoin.sol#184)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
changeOwnership(address) should be declared external:
    - Owned.changeOwnership(address) (kucoin.sol#15-17)
changeFreezeStatus(bool) should be declared external:
    - Freeze.changeFreezeStatus(bool) (kucoin.sol#28-30)
balanceOf(address) should be declared external:
    - BEP20.balanceOf(address) (kucoin.sol#57)
transferFrom(address,address,uint256) should be declared external:
    - BEP20.transferFrom(address,address,uint256) (kucoin.sol#68-76)
approve(address,uint256) should be declared external:
    - BEP20.approve(address,uint256) (kucoin.sol#78-82)
allowance(address,address) should be declared external:
    - BEP20.allowance(address,address) (kucoin.sol#84-86)
timelock(address,uint256,uint256) should be declared external:
    - TimeLock.timelock(address,uint256,uint256) (kucoin.sol#131-139)
release() should be declared external:
    - TimeLock.release() (kucoin.sol#141-155)
lockedAccountDetails(address) should be declared external:
    - TimeLock.lockedAccountDetails(address) (kucoin.sol#157-173)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-
that-could-be-declared-external
```

Mythril

```
myth a --rpc 127.0.0.1:8545 -a 0x27a1b3FDA67b48d09986306B397d390f011E0156
The analysis was completed successfully. No issues were detected.
```

THEO

```
theo --rpc-http http://127.0.0.1:8545
The account's private key (input hidden)
>
Contract to interact with
> 0x27a1b3FDA67b48d09986306B397d390f011E0156
Scanning for exploits in contract: 0x27a1b3FDA67b48d09986306B397d390f011E0156
Connecting to HTTP: http://127.0.0.1:8545.
No exploits found. You're going to need to load some exploits.

Tools available in the console:
- `exploits` is an array of loaded exploits found by Mythril or read from a file
- `w3` an initialized instance of web3py for the provided HTTP RPC endpoint
- `dump()` writing a json representation of an object to a local file

Check the readme for more info:
https://github.com/cleanunicorn/theo

Theo version v0.8.2.
```

SOLHINT LINTER

```
kucoin.sol
    6:1   error    Compiler version 0.8.4 does not satisfy the ^0.5.8 semver requirement
compiler-version
   10:9   warning  Provide an error message for require
reason-string
   52:5   warning  Explicitly mark visibility of state
state-visibility
   53:5   warning  Explicitly mark visibility of state
state-visibility
   57:45  warning  Visibility modifier must be first in list of modifiers
visibility-modifier-order
   61:9   warning  Provide an error message for require
reason-string
```

```
   70:9   warning  Provide an error message for require
reason-string
   84:63  warning  Visibility modifier must be first in list of modifiers
visibility-modifier-order
  100:9   warning  Error message for require is too long
reason-string
  105:9   warning  Error message for require is too long
reason-string
  111:94  warning  Code contains empty blocks
no-empty-blocks
  129:5   warning  Explicitly mark visibility of state
state-visibility
  137:61  warning  Avoid to make time-based decisions in your business logic
not-rely-on-time
  147:13  warning  Error message for require is too long
reason-string
  147:21  warning  Avoid to make time-based decisions in your business logic
not-rely-on-time
  152:13  warning  Possible reentrancy vulnerabilities. Avoid state changes after transfer
reentrancy
  163:28  warning  Avoid to make time-based decisions in your business logic
not-rely-on-time
  180:5   warning  Explicitly mark visibility in function (Set ignoreConstructors to true if
using solidity >=0.7.0)  func-visibility
  191:38  error    Use double quotes for string literals
quotes
  196:38  error    Use double quotes for string literals
quote
```

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.

Disclaimer

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

Closing Summary

In this report, we have considered the security of the KUCOIN LAUNCHPAD platform. We performed our audit according to the procedure described above.

Several issues of medium, low, and informational severity were discovered during the audit. In the end, all of the issues were resolved by the Auditee.

QuillAudits

Canada, India, Singapore and United Kingdom

audits.quillhash.com

audits@quillhash.com