QuillAudits

Audit Report
May, 2021

SUPER LAUNCHER

Contents

Introduction

This Audit Report highlights the overall security of the SuperLauncher Smart Contract. With this report, we have tried to ensure the reliability of their smart contract by a complete assessment of their system's architecture and the smart contract codebase.

Auditing Approach and Methodologies applied

The Quillhash team has performed thorough testing of the project starting with analysing the code design patterns in which we reviewed the smart contract architecture to ensure it is structured and practices safe use of third party smart contracts and libraries.

Our team then performed a formal line by line inspection of the Smart Contract to find any potential issue like race conditions, transaction-ordering dependence, timestamp dependence, and denial of service attacks.

In the Unit testing Phase, we coded/conducted Custom unit tests written for each function in the contract to verify that each function works as expected. In Automated Testing, we tested the Smart Contract with tools developed in-house to identify vulnerabilities and security flaws.

The code was tested in collaboration with multiple team members and this included -

- Testing the functionality of the Smart Contract to determine that proper logic was followed throughout the process.
- Analysing the complexity of the code by thorough, manual review of the code, line-by-line.
- Deploying the code on testnet using multiple clients to run live tests
- Analyzing failure preparations to check how the Smart Contract performs in case of bugs and vulnerabilities.
- Checking whether all the libraries used in the code are on the latest version.
- Analyzing the security of the on-chain data.

Audit Details

Project Name: Superlauncher
Website/Etherscan Code: Github
(154a063eaf73c76c678e528f2dc8635fca5cb78f)
Languages: Solidity (Smart contract), Javascript (Unit Testing)
Platforms and Tools: Remix IDE, Truffle, Truffle Team, Ganache, Slither, Surya

Summary of SuperLauncher Smart Contract

QuillAudits conducted a security audit of a smart contract of SuperLauncher. SuperLauncher contract is used to create the Campaign and vesting for campaign tokens (BEP20) as well as the raised BNB funds.

Name: SuperLauncher
Symbol: LAUNCH

And some advanced features other than essential functions.

- Vesting
- Campaign creation

Audit Goals

The focus of the audit was to verify that the smart contract system is secure, resilient and working according to its specifications. The audit activities can be grouped into the following three categories:

Security
Identifying security related issues within each contract and the system of contract.

Sound Architecture
Evaluation of the architecture of this system through the lens of established smart contract best practices and general software best practices.

Code Correctness and Quality

A full review of the contract source code. The primary areas of focus include:

- Accuracy
- Readability
- Sections of code with high complexity
- Quantity and quality of test coverage

Security Level references

Every issue in this report was assigned a severity level from the following:

High level severity issues

Issues on this level are critical to the smart contract's performance/ functionality and should be fixed before moving to a live environment.

Medium level severity issues

Issues on this level could potentially bring problems and should eventually be fixed.

Low level severity issues

Issues on this level are minor details and warnings that can remain unfixed but would be better fixed at some point in the future.

Number of issues per severity

	Low	Medium	High	Recommendations
Open	0	0	0	0
Closed	2	0	0	0

Manual Audit

High level severity issues

No high severity issues

Medium level severity issues

No medium severity issues

Low level severity issues

1. **Unused Return**

   Campaign.addAndLockLP (Campaign.sol) does not use the value returned by external calls:

   -ERC20(address(token)).approve(lpRouterAddress,lpTokenQty) (Campaign.sol)

   Use return value of external calls

   Status: Fixed

2. **Variables that can be declared external**

   Campaign.buyTokens (Campaign.sol) should be declared external
   Campaign.addAndLockLP (Campaign.sol) should be declared external
   Campaign.getPoolLP (Campaign.sol) should be declared external
   Campaign.finishUp (Campaign.sol) should be declared external
   Campaign.withdrawLP (Campaign.sol) should be declared external
   Campaign.refund (Campaign.sol) should be declared external
   Campaign.appendWhitelisted (Campaign.sol) should be declared external
   Campaign.removeWhitelisted (Campaign.sol) should be declared external
   Campaign.setCancelled (Campaign.sol) should be declared external

   Note: It is advice to update state variables before external calls, even though external calls are within smart contracts of yours, or take all the precautions of transaction failure and use exception handling in external calls.

   Status: Fixed

Automated Testing

Slither Tool Result

```
Reference: https://github.com/trailofbits/slither/wiki/Detectors-Documentation#different-pragma-directives-are-used
INFO:Detectors:
Campaign.buyTokens (Campaign.sol#295-319) should be declared external
Campaign.addAndLockLP (Campaign.sol#327-346) should be declared external
Campaign.getPoolLP (Campaign.sol#353-355) should be declared external
Campaign.finishUp (Campaign.sol#400-425) should be declared external
Campaign.withdrawLP (Campaign.sol#469-470) should be declared external
Campaign.refund (Campaign.sol#482-497) should be declared external
Campaign.appendWhitelisted (Campaign.sol#510-525) should be declared external
Campaign.removeWhitelisted (Campaign.sol#532-541) should be declared external
Campaign.setCancelled (Campaign.sol#650-664) should be declared external
Reference: https://github.com/trailofbits/slither/wiki/Detectors-Documentation#public-function-that-could-be-declared-as-external
INFO:Detectors:
Detected issues with version pragma in Campaign.sol:
        - pragma solidity^0.5.16 (Campaign.sol#2): it allows old versions
        - pragma solidity^0.5.16 (Interfaces.sol#15): it allows old versions
Reference: https://github.com/trailofbits/slither/wiki/Detectors-Documentation#incorrect-version-of-solidity
INFO:Detectors:
Parameter '_token' of Campaign.initialize (Campaign.sol#237) is not in mixedCase
Parameter '_campaignOwner' of Campaign.initialize (Campaign.sol#238) is not in mixedCase
Parameter '_stats' of Campaign.initialize (Campaign.sol#239) is not in mixedCase
Parameter '_dates' of Campaign.initialize (Campaign.sol#240) is not in mixedCase
Parameter '_buyLimits' of Campaign.initialize (Campaign.sol#241) is not in mixedCase
Parameter '_access' of Campaign.initialize (Campaign.sol#242) is not in mixedCase
Parameter '_liquidity' of Campaign.initialize (Campaign.sol#243) is not in mixedCase
Parameter '_burnUnSold' of Campaign.initialize (Campaign.sol#244) is not in mixedCase
Parameter '_amount' of Campaign.withdrawLP (Campaign.sol#469) is not in mixedCase
Parameter '_user' of Campaign.getTotalTokenPurchased (Campaign.sol#505) is not in mixedCase
Parameter '_addresses' of Campaign.appendWhitelisted (Campaign.sol#510) is not in mixedCase
Parameter '_addresses' of Campaign.removeWhitelisted (Campaign.sol#532) is not in mixedCase
Parameter '_address' of Campaign.checkWhiteList (Campaign.sol#549) is not in mixedCase
Parameter '_to' of Campaign.sendTokensTo (Campaign.sol#570) is not in mixedCase
Parameter '_amt' of Campaign.getFeeAmt (Campaign.sol#580) is not in mixedCase
Parameter '_bobInvestment' of Campaign.calculateTokenAmount (Campaign.sol#630) is not in mixedCase
Parameter '_periods' of Campaign.setupVestingMode (Campaign.sol#704) is not in mixedCase
Parameter '_percents' of Campaign.setupVestingMode (Campaign.sol#704) is not in mixedCase
```

```
INFO:Detectors:
Campaign.lpPool (Campaign.sol#123) is never initialized. It is used in:
        - getPoolLP (Campaign.sol#353-355)
        - recoverUnspentLP (Campaign.sol#360-391)
Reference: https://github.com/trailofbits/slither/wiki/Detectors-Documentation#uninitialized-state-variables
INFO:Detectors:
Contract locking ether found in Campaign.sol:
        Contract Campaign has payable functions:
         - buyTokens (Campaign.sol#295-319)
        But does not have a function to withdraw the ether
Reference: https://github.com/trailofbits/slither/wiki/Detectors-Documentation#contracts-that-lock-ether
INFO:Detectors:
Reentrancy in Campaign.buyTokens (Campaign.sol#295-319):
        External calls:
        - require(bool,string)(checkQualifyingTokens(msg.sender),Insufficient LAUNCH tokens to qualify) (Campaign.sol#298)
        State variables written after the call(s):
        - collectedBNB (Campaign.sol#310)
Reference: https://github.com/trailofbits/slither/wiki/Detectors-Documentation#reentrancy-vulnerabilities-1
INFO:Detectors:
remainTime in Campaign.isVestingClaimable (Campaign.sol#758) is a local variable never initialiazed
Reference: https://github.com/trailofbits/slither/wiki/Detectors-Documentation#uninitialized-local-variables
INFO:Detectors:
Campaign.addAndLockLP (Campaign.sol#327-346) does not use the value returned by external calls:
        -ERC20(address(token)).approve(lpRouterAddress,lpTokenQty) (Campaign.sol#339)
Reference: https://github.com/trailofbits/slither/wiki/Detectors-Documentation#unused-return
INFO:Detectors:
Reentrancy in Campaign.addAndLockLP (Campaign.sol#327-346):
        External calls:
        - lpRouterAddress = fact.getLpRouter() (Campaign.sol#338)
        - ERC20(address(token)).approve(lpRouterAddress,lpTokenQty) (Campaign.sol#339)
        State variables written after the call(s):
        - unlockDate (Campaign.sol#343)
Reentrancy in Campaign.buyTokens (Campaign.sol#295-319):
        External calls:
        - require(bool,string)(checkQualifyingTokens(msg.sender),Insufficient LAUNCH tokens to qualify) (Campaign.sol#298)
```

Implementation Recommendations

Parameter '_token' of Campaign.initialize (Campaign.sol#237) is not in mixedCase

Parameter '_campaignOwner' of Campaign.initialize (Campaign.sol#238) is not in mixedCase

Parameter '_stats' of Campaign.initialize (Campaign.sol#239) is not in mixedCase

Parameter '_dates' of Campaign.initialize (Campaign.sol#240) is not in mixedCase

Parameter '_buyLimits' of Campaign.initialize (Campaign.sol#241) is not in mixedCase

Parameter '_access' of Campaign.initialize (Campaign.sol#242) is not in mixedCase

Parameter '_liquidity' of Campaign.initialize (Campaign.sol#243) is not in mixedCase

Parameter '_burnUnSold' of Campaign.initialize (Campaign.sol#244) is not in mixedCase

Parameter '_amount' of Campaign.withdrawLP (Campaign.sol#469) is not in mixedCase

Parameter '_user' of Campaign.getTotalTokenPurchased (Campaign.sol#505) is not in mixedCase

Parameter '_addresses' of Campaign.appendWhitelisted (Campaign.sol#516) is not in mixedCase

Parameter '_addresses' of Campaign.removeWhitelisted (Campaign.sol#532) is not in mixedCase

Parameter '_address' of Campaign.checkWhiteList (Campaign.sol#549) is not in mixedCase

Parameter '_to' of Campaign.sendTokensTo (Campaign.sol#570) is not in mixedCase

Parameter '_amt' of Campaign.getFeeAmt (Campaign.sol#586) is not in mixedCase

Parameter '_bnbInvestment' of Campaign.calculateTokenAmount (Campaign.sol#630) is not in mixedCase

Parameter '_periods' of Campaign.setupVestingMode (Campaign.sol#704) is not in mixedCase

Parameter '_percents' of Campaign.setupVestingMode (Campaign.sol#704) is not in mixedCase

Parameter '_index' of Campaign.isVestingClaimable (Campaign.sol#750) is not in mixedCase

Parameter '_index' of Campaign.claimVestedTokens (Campaign.sol#770) is not in mixedCase

Parameter '_index' of Campaign.claimVestedBnb (Campaign.sol#794) is not in mixedCase

Parameter '_user' of Campaign.getNextVestingClaim (Campaign.sol#814) is not in mixedCase

Disclaimer

Quillhash audit is not a security warranty, investment advice, or an endorsement of the SuperLauncher contract. Securing smart contracts is a multistep process, therefore running a bug bounty program as a complement to this audit is strongly recommended.

Summary

The use case of the smart contract is very well designed and Implemented. Overall, the code is written and demonstrates effective use of abstraction, separation of concerns, and modularity. The **SuperLauncher** development team demonstrated high technical capabilities, both in the design of the architecture and in the implementation.

All bugs, suggestions and recommendations have been considered by the SuperLauncher team and some of the issues they will handle on their own as those issues or calls have been handled by the owner.

SUPER LAUNCHER

QuillAudits

Canada, India, Singapore and United Kingdom

audits.quillhash.com

hello@quillhash.com