ALPHA APE LLC

New Community Luna
cLuna

Audit Report
May.19.2022

Table of Contents

SUMMARY OF AUDIT

DETAILS OF AUDITED PROJECT

Audited Project:	New Community Luna – (cLuna)
Source Code:	https://bscscan.com/address/0x8a0c6c59a80292b1a7fc8770bfbb1e27801c5e9c#code
Solidity File:	cLuna.sol
Security Audit Date:	May. 19 - 2022
Revisions:	None
Auditing Methods:	Automatic review + Manual review

AUDITING METHODS AND COVERING SECTORS

Evaluation objective for the security audit:

- Quality of smart contracts code
- Issues and vulnerabilities with security
- Documentation, project specifics and commenting on smart contract
- Correctness of specifications regarding the use-case
- Following the best practices on smart contract

Audit covers these sectors of smart contract for possible vulnerabilities, issues and recommendations for better practices in case of severe or medium issues:

- Dependence Transaction Order
- Single and Cross-Function Reentrancy
- Time Dependency
- Integer Overflow
- Integer Underflow
- Mishandled exceptions and call stack limits
- Unsafe external calls
- Number rounding errors
- Malicious contract behaviour and abusable functions
- Possible DoS vulnerabilities
- Insufficient gas issues
- Logical oversights
- Access control
- Centralization of power
- Logic-Specification
- Contradiction
- Functionality duplication

The code review conducted for this audit follows the following structure:

1. Review of the specifications, documentation and commenting provided by the project owners regarding the functionality of the smart contract
2. Automated analysis of the smart contract followed by manual, line-by-line analysis of the smart contract
3. Assessment of smart contracts correctness regarding the documentation and commenting compared to functionality
4. Assessment of following the best practices
5. Recommendations for better practises in case severe or medium vulnerabilities

SMART CONTRACT DETAILS

Contract ID	0x8A0C6c59a80292b1A7fc8770bFbB1E27801C5E9c
Blockchain:	Binance Smart Chain
Language Used:	Solidity
Compiler Version:	v0.6.12+commit.27d51765
Bscscan Verification:	2022-05-13
Type of Smart Contract:	BEP20 Token
Libraries Used:	
Optimization Enabled:	Yes with 200 runs

SUMMARY OF AUDIT RESULTS

Alpha Ape Network smart contract audit for New Community Luna – (cLuna) is marked as **PASSED** result without severe issues on the logic and functions of the contract. Review of the documentation and description of the projects use-case and the smart contract follows the line of good practices. Clean and default commented contract without hidden abusable functions. Contract is renounced and no contract functions can be changed.

SEVERITY OF RISKS AND VULNERABILITIES

LOW-SEVERITY	MEDIUM-SEVERITY	HIGH-SEVERITY
2	**0**	**0**

REPORTED VULNERABILITIES AND ISSUES

LEVEL OF SEVERITY	DESCRIPTION	FILE	CODELINES AFFECTED
LOW-SEVERITY	A floating pragma is set.	cLuna.sol	L: 32 C: 0
LOW-SEVERITY	State variable visibility is not set.	cLuna.sol	L: 743 C: 9

FINDINGS IN-DEPTH

LOW-SEVERITY A FLOATING PRAGMA IS SET

```
30  */
31
32  pragma solidity ^0.6.12;
33  // SPDX-License-Identifier: Unlicensed
34  interface IERC20 {
```

Analysis: The current pragma Solidity directive is ""^0.6.12"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Possible vulnerability: Contracts should be deployed with the same compiler version and flags that they have been tested with thoroughly. Locking the pragma helps to ensure that contracts do not accidentally get deployed using, for example, an outdated compiler version that might introduce bugs that affect the contract system negatively.

Recommendation: Lock the pragma version and also consider known bugs (https://github.com/ethereum/solidity/releases) for the compiler version that is chosen.
Pragma statements can be allowed to float when a contract is intended for consumption by other developers, as in the case with contracts in a library or EthPM package. Otherwise, the developer would need to manually update the pragma in order to compile locally.

cLuna.sol L: L: 32 C: 0

LOW-SEVERITY STATE VARIABLE VISIBILITY IS NOT SET

```
741  address public immutable uniswapV2Pair;
742
743  bool inSwapAndLiquify;
744  bool public swapAndLiquifyEnabled = true;
745
```

Analysis: It is best practice to set the visibility of state variables explicitly. The default visibility for "inSwapAndLiquify" is internal. Other possible visibility settings are public and private and should always be stated. This is not an vulnerability risk, but an improper adherence to coding standards

Possible vulnerability: Labeling the visibility explicitly makes it easier to catch incorrect assumptions about who can access the variable.

Recommendation: Variables can be specified as being public, internal or private. Explicitly define visibility for all state variables for understanding the calls from contract better and adhere to coding standards regarding best practices.

cLuna.sol L: L: 743 C: 9

DOCUMENTATION AND COMMENTING

The code has a decent amount of comments and documentation. Improving stage of commenting on smart contracts, and cleaning the commenting, helps userbase to understand all the functionalities and use-case of the contract. Functions are marked in understandable way.

CORRECTNESS OF SPECIFICATIONS

Smart contract follows the functionality that is stated in the documentation and description of the contract. The use-case is also in line what is described about the project. Verified as an community project without centralized project funds.

FOLLOWING THE BEST PRACTICES

The smart contract follows the best practices in majority. Minor parts that do not follow the best practices, or is affected by possible vulnerabilities, do not rise fear in malicious use of the contract or severe issues in projects use-case. Low-level severities that are noted in the audit are only improper adherence to coding standards, than security vulnerabilities.

HISTORY OF REVISIONS, FUNCTIONS AND VARIABLES

HISTORY OF REVISIONS

Initial audit was performed May. 19-2022. No further revision history.

Team has been provided recommendations for better practices for revision. There is no severe or medium notifications or recommendations for the revision. Contract is deployed and immutable and the possible vulnerabilities has been stated in the security audit. Project is stated to be community driven and security auditor has validated the use-case of the contract to follow its description the best possible way.