Smart Contract Audit Conclusion

by Vinh Le

2th October 2022

This is the review of Aliroody ’s Fundraiser-finance repo. No critical bug found, but there exists several issues.

Findings

ID	Severity	Subject
CVF-1	High	Improper approach
CVF-2	Medium	Different pragma directives are used
CVF-3	Medium	Improper Solidity version
CVF-4	Medium	Improper approach
CVF-5	Minor	Too many digits
CVF-6	Minor	Block timestamp
CVF-7	Minor	Function declaration

Contents

1. Document properties

Version

Version	Date	Author	Description
0.1	Aug. 24, 2022	Vinh Le	Initial Draft

Contact

Vinh Le

minhthangminh1992@gmail.com

2. Introduction

The following document provide the results of the audit performed by Vinh Le Consulting. The audit goal is a general review of the smart contracts structure, critical/major bugs detection and issuing the general recommendations.

We have audited the fundraiser-finance Concretely, the following file was audited:

- src/Badge.sol
- src/CharityFactory.sol

3. Detailed Results

3.1 CVF-1 Improper approach

- Severity Minor
- Category Suboptimal

Description Benign reentrancy vulnerability

Recommendation Apply the `check-effects-interactions`pattern

Listing 1: Improper approach

External calls:

- USDC_ADDRESS.transferFrom(address(this),charity.beneficiary,charity.usdcRaised)

(src/CharityFactory.sol#192)

- (success) = address(charity.beneficiary).call{value: charity.ethRaised}()

(src/CharityFactory.sol#195)

External calls sending eth:

- (success) = address(charity.beneficiary).call{value: charity.ethRaised}()

(src/CharityFactory.sol#195)

Event emitted after the call(s):

- CloseCharity(charityId,charity.status) (src/CharityFactory.sol#200)

Reentrancy in CharityFactory.withdrawContribution(uint256) (src/CharityFactory.sol#147-167)

3.2 CVF-2 Different pragma directives are used

- Severity Minor
- Category Suboptimal

Description Different versions of Solidity are used throughout the OpenZepellin dependencies and src/Badge.sol, src/Charity.sol

Recommendation Use one Solidity version.

Listing 2: Different pragma directives are used

Version used: ['>=0.4.22<0.9.0', '>=0.5.0', '>=0.6.0<0.9.0', '^0.8.0', '^0.8.1', '^0.8.13', '^0.8.9']

3.3 CVF-3 Improper Solidity version

- Severity Minor
- Category Suboptimal

Description Pragma version^0.8.13 (src/CharityFactory.sol#2) necessitates a version too recent to betrusted.

Recommendation Consider deploying with 0.6.12/0.7.6/0.8.7

Listing 3: Improper Solidity version

```
2       pragma solidity ^0.8.13;
```

3.4 CVF-4 Improper approach

- Severity Minor
- Category Suboptimal

Description Low level call in Cutie.withdraw(). The use of low-level calls is error-prone. Low-level calls do not check for code existence or call success.

Recommendation Avoid low-level calls. Check the call success. If the call is meant for a contract, check for code existence.

Listing 4: Improper approach

```
(success) = address(charity.beneficiary).call{value: charity.ethRaised}() (src/CharityFactory.sol#195)
```

3.5 CVF-5 Too many digits

- Severity Minor
- Category Too many digits

Description The declared number uses literals with too many digits. Literals with many

digits are difficult to read and review.

Recommendation Use exponential suffix

Listing 5: Too many digits

```
creationTimestamp = 1000000 (test/CharityFactory.t.sol#20)
```

3.6 CVF-6 Block timestamp

- Severity Minor
- Category Assembly misuse

Description Dangerous usage of block.timestamp. block.timestamp can be manipulated by

miners.

Recommendation Avoid relying on block.timestamp.

Listing 6: Block timestamp

```
 require(bool,string)(block.timestamp < charity.endPeriod,Cannot donate to closed charity)
(src/CharityFactory.sol#103)
```

3.7 CVF-7 Function declaration

- Severity Minor
- Category Suboptimal

Description public functions that are never called by the contract should be declared external to save gas.

Recommendation mint(address,uint256,uint256,uint256) should be declared external

Listing 7: Function declaration

```
Badge.mint(address,uint256,uint256,uint256) (src/Badge.sol#33-44)
```