G_0 Group

Security Review of Nectar DAO

October 12, 2019

Overview

G0 Group was engaged to perform a security review of the DAOStack extensions (schemes) to be used by Nectar DAO (a DAO which will govern the Deversifi DEX). G0 Group was contracted for a two person-week effort to that end, followed by a one person-week review of fixes. Additionally, G0 Group has previously conducted a security review of the DAOStack codebase: DAOStack review report. The primary subjects of this review were the new extensions which implement: Nectar DAO's reputation allocation and allow DAOs to generically call functions on a given contract (which is immutable following initialization); as well as their interaction with the existing DAOStack code. Together these components form Nectar DAO's particular DAOStack deployment. This review was initially performed on https://github.com/daostack/arc/blob/0.0.1-rc.24/.

Files in Scope

```
contracts/
    schemes/
        ContinuousLockingToken4Reputation.sol
        ReputationFromToken.sol
        GenericScheme.sol
    test/
        NectarRepAllocation.sol
```

Result Summary

During the course of this review, 3 issues were discovered and reported. One of these issues posed a direct security threat, the rest concerned potential issues and best practice. All security issues reported have been remediated, and are not present in https://github.com/daostack/arc/releases/tag/0.0.1-rc.31.

No further issues were discovered.

Issues

1. Attacker can reduce their locking period at will due to an overflow issue

Type: *security /* ***Severity:*** *major*

In `ContinuousLockingToken4Reputation.sol`: `line 222` in `extendLocking` can overflow, then the check on `line 223` will be passed and `line 237` will overflow too, allowing an attacker to reduce the locking period instead of extending it.

Fix Description:

Issue was addressed in https://github.com/daostack/arc/pull/670 by using a SafeMath library for math operations.

2. Fragile code segment in ContinuousLockingToken4Reputation could lead to the introduction of a vulnerability in a future update

Type: *security /* ***Severity:*** *potential issue (fragile code)*

In the `redeem` function of `ContinuousLockingToken4Reputation.sol` there should be a check to ensure the locker exists. Even though the absence of the check is not exploitable right now, including an existence check would make the code more robust by enforcing what should be an invariant.

Fix Description:

Issue was addressed in https://github.com/daostack/arc/pull/670 by adding an existence check.

3. Unnecessary use of public visibility on many functions

Type: *note*

This is a not a security concern; however, in many cases a `public` visibility is used where `external` would suffice. In favor of code clarity and potential gas savings, we recommend setting `external` on all functions that don't need to be called internally.