Aragon Minters
Smart Contract Audit

coinspect

Smart Contract Audit
Aragon Minters

Prepared for Aragon Association • January 2021

V210121

1. Executive Summary

In January 2021, Aragon Association engaged Coinspect to perform a source code review of two new smart contracts that implement ANT minting. The goal of the project was to evaluate the security of the smart contracts.

The assessment was conducted on the following pull requests:

> https://github.com/aragon/aragon-network-token/pull/37
> https://github.com/aragon/aragon-network-token/pull/38

The two pull requests introduce two new contracts: `ANTv2MultiMinter` and `ANJNoLockMinter`.

The correctness of the two new contracts was verified during the assessment. No security issues were identified.

Although the contracts were determined to be correct, the pull requests do not include unit tests and it is recommended to implement tests for the new contracts.

2. Introduction

The audit started on January 18th and was conducted on the following pull requests:

https://github.com/aragon/aragon-network-token/pull/37
https://github.com/aragon/aragon-network-token/pull/38

This corresponds to the following two smart contracts, shown here with their sha256sum:

```
0f3b7bca854a3197e9404f8db4e8d024a400050105d39c6b0aa706afb568f385  ANTv2MultiMinter.sol
8f162f80b08770cf9bb9b6e38fa49b016049c17fe0a6beb46a127f6385171252  ANJNoLockMinter.sol
```

3. Assessment

The two new contracts `ANTv2MultiMinter` and `ANJNoLockMinter` are specified to be compiled with Solidity version 0.5.17. This is the latest version of the 0.5.x series and it is considered stable and safe.

The contracts compile without warnings and all tests pass. However, the repository doesn't include any new tests for the two new contracts, and although the contracts are very simple and their correctness can be quickly verified by carefully reading the code, it is recommended to add tests.

3.1. ANTv2MultiMinter

The `ANTv2MultiMinter` contract, after being set as the minter of the `ANTv2` contract, allows for different contracts to mint `ANT`. This is necessary to allow contracts that will automatically mint `ANT` (for example contracts for the `ANJ` merge conversion like the new `ANJNoLockMinter`) as well as allowing for future arbitrary or automatic minting by the Aragon Network DAO.

The `ANTv2MultiMinter` contract is constructed with an *owner* and a reference to an `ANTv2` contract of which it is supposed to be the minter. It keeps a mapping of *minters*, which are addresses allowed to mint `ANTv2` tokens by calling the `mint` function in `ANTv2MultiMinter`. Only the owner or minters added by the owner are allowed to call the `mint` function to mint `ANTv2` tokens. Only the owner of `ANTv2MultiMinter` can add or remove minters by calling `addMinter` and `removeMinter`, and can also transfer ownership of the `ANTv2MultiMinter` by calling `changeOwner` and set the minter of the `ANTv2` contract from the `ANTv2MultiMinter` itself to a new address by calling `changeMinter`. The function `addMinter` emits the event `AddedMinter(address indexed minter)`, `removeMinter` emits `RemovedMinter(address indexed minter)` and `changeOwner` emits `ChangedOwner(address indexed newOwner)`.

3.2. ANJNoLockMinter

The `ANJNoLockMinter` contract is intended to be added as minter in the `ANTv2MultiMinter` contract and implements functions for minting `ANT` in exchange for burning `ANJ` at a 0.015 `ANT` per `ANJ` as approved by `ANT` holders.

The `ANJNoLockMinter` contract is very similar to the `ANTv2Migrator` contract that was implemented for migrating `ANTv1` to `ANTv2`. It is constructed with the addresses of the `ANTv2MultiMinter`, the `ANTv2` contract and the `ANJ` contract, and it is assumed to be registered as a minter in the `ANTv2MultiMinter` contract. Users with a positive ANJ balance can use this contract to convert their ANJ to ANT. The users have three options:

- Calling the function `migrate` with the desired amount to be converted as an argument.
- Calling the function `migrateAll` which is equivalent to calling `migrate` with the user's total balance.

- Calling the `approveAndCall` function in the `ANJ` contract with the desired amount to be converted and the `ANJNoLockMinter` contract address as arguments.

As a result, the desired number of ANJ tokens are burned (transferred to the address `0x000000000000000000000000000000000000dEaD`) and the corresponding number of ANT tokens are minted for the user.

4. Disclaimer

The information presented in this document is provided as is and without warranty. Vulnerability assessments are a “point in time” analysis and as such it is possible that something in the environment could have changed since the tests reflected in this report were run. This report should not be considered a perfect representation of the risks threatening the analysed system, networks and applications.