AgoraToken - Smart Contract Audit

By: Sam Ngyuen <sam.e.nguyen@hotmail.com>
Latest revision: March 21st 2022

Table of Contents

Overview

A modified mintable ERC20 token and a Staking contract. The staking logic should require a locking period between 30 and 360 days, with the APY being governed by a curve described in https://www.desmos.com/calculator/klc2xjja3a

Dates

- **March 19th*: Start date
- **March 21st*: First report version

Process

This document, and all suggestions detailed here, is meant to be scrutinized and discussed between both parties, in an ongoing collaborative process after the first report delivery. Each suggestion may not be needed due to contextual reasons, or limited understanding of external scope by the auditor.

Coverage

The code in-scope for the review was provided in private, and accessible in the private repo where this audit was being conducted

Areas of Concern

The investigation focused on the following:

- Looking for attack vectors that could impact the intended behaviour of the smart contract
- Checking test coverage, particularly for potentially dangerous scenarios and edge-cases
- Interaction with 3rd party contracts
- Use of solidity best practices
- Permissions and roles after deploy script is executed

Findings

Each issue has an assigned severity:

- **Minor** issues are subjective in nature. They are typically suggestions around best practices or readability. Code maintainers should use their judgement as to whether to address such issues.
- **Gas** issues are related to gas optimizations.
- **Medium** issues are objective in nature but are not security vulnerabilities. These should be addressed unless there is a clear reason not to.

- **High** issues are security vulnerabilities that may not be directly exploitable or may require certain conditions in order to be exploited. All major issues should be addressed.
- **Critical** issues are directly exploitable security vulnerabilities that need to be fixed.
- **Note** issues are notes from the auditor, such as more subjective improvement suggestions, lack of context that limited the auditor's ability to evaluate the work, or others. These are mostly informational, but may result in additional discussion (e.g.: provide additional context to the auditor)

Progress

This table helps tracks the auditor’s progress through each section of the codebase, and is not indicative of final results.

contract	progress	status
GAERC20	ok	ok
AgoraToken	ok	ok
Staking	ok	ok

Findings Summary

Findings are listed in chronological order of discovery.

title	severity	status
N1. Should allowances() return 0 for globally approved spenders?	**Note**	

Detailed Findings

N1. Should allowances() return 0 for globally approved spenders?

Note

`GAERC20` allows setting globally approved spends. However, when calling `allowance(owner, globallyApprovedSpender)` , the result will still be governed by the individual `_allowances` mapping. Should this value perhaps return `type(uint256).max` when the spender is globally approved?

The current approach may cause issues if some automatic tooling or 3rd party contract relies on calling `allowance` to check if an `approval` is required before issuing a transaction, which may or may not be a concern for this token.