SOLIDIFIED

Audit Report for API3 DAO - April 15, 2021

Summary

Audit Report prepared by Solidified covering the API3 DAO smart contracts.

Process and Delivery

Three (3) independent Solidified experts performed an unbiased and isolated audit of the code below. The final debrief took place on March 22, 2021, and the results are presented here.

Audited Files

The source code has been supplied in the form of a GitHub repository:

https://github.com/api3dao/api3-dao/tree/solidified

Commit number: `0f3347673210129a28cfd5a0e27ef7bf1285845d`

UPDATE: Fixes have been supplied in the default branch of the repository:

https://github.com/api3dao/api3-dao

Commit number: `5fb38bc91ae832f7b94f9630a9d41d15a2c97d27`

Note, that the content of the directories `auxiliary` and `mock` have been explicitly excluded from the code for the audit by team.

The scope of the audit was limited to the following files:

```
api3-voting/contracts
├── Api3Voting.sol
└── interfaces
   └── IApi3Pool.sol


dao/contracts
├── Api3Template.sol
└── interfaces
    └── IApi3Pool.sol

pool/contracts
├── Api3Pool.sol
├── ClaimUtils.sol
├── DelegationUtils.sol
├── GetterUtils.sol
├── RewardUtils.sol
├── StakeUtils.sol
├── StateUtils.sol
├── TimelockUtils.sol
├── TransferUtils.sol
└── interfaces
    ├── IApi3Pool.sol
    ├── IClaimUtils.sol
    ├── IDelegationUtils.sol
    ├── IGetterUtils.sol
    ├── IRewardUtils.sol
    ├── IStakeUtils.sol
    ├── IStateUtils.sol
    ├── ITimelockUtils.sol
    └── ITransferUtils.sol
```

Intended Behavior

The smart contracts implement DAO as an Aragon voting app and an associated staking pool.

Code Complexity and Test Coverage

Smart contract audits are an important step to improve the security of smart contracts and can find many issues. However, auditing complex codebases has its limits and a remaining risk is present (see disclaimer).

Users of a smart contract system should exercise caution. In order to help with the evaluation of the remaining risk, we provide a measure of the following key indicators: **code complexity**, **code readability**, **level of documentation**, and **test coverage**.

Note, that high complexity or lower test coverage does equate to a higher risk. Certain bugs are more easily detected in unit testing than a security audit and vice versa. It is, therefore, more likely that undetected issues remain if the test coverage is low or non-existent.

Criteria	Status	Comment
Code complexity	Medium	-
Code readability and clarity	High	-
Level of Documentation	Medium-High	-
Test Coverage	High	-

Issues Found

Solidified found that the API3 DAO contracts contain no critical issue, 1 major issue, no minor issues, in addition to 1 warning and 1 informational note.

We recommend all issues are amended, while the notes are up to the team's discretion, as they refer to best practices.

Issue #	Description	Severity	Status
1	StakeUtils.sol: Contract users can potentially be indefinitely locked out of unstaking their tokens	Major	Resolved
2	Non-Existent Test Coverage for DAO package	Warning	Resolved
3	Api3Template.sol: Element count discrepancy for _votingSettings array	Note	Resolved

Critical Issues

No critical issues have been found.

Major Issues

1. `StakeUtils.sol`: Contract users can potentially be indefinitely locked out of unstaking their tokens

Users currently use function `scheduleUnstake()` to schedule unstaking their tokens. This function calls `getValue()`, which in turn searches the given array for the checkpoint at the current block.

If either of the `totalStaked` or `totalShares` arrays were to eventually grow to a sufficiently large size, this binary search will exceed the current block gas limit, and `getValue()`/`getValueAt()` will always fail. This will result in `scheduleUnstake()` always failing, and in turn all contract users will be indefinitely prevented from unstaking their tokens.

Note
Functions `StakeUtils.stake()` / `RewardUtils.payReward()` / `CalimUtils.payOutClaim()` can all potentially suffer from the same vulnerability, albeit to a lesser extent of severity.

Recommendation

Place a hard cap on the size of the aforementioned arrays that will not exceed the current block gas limit. Also, consider doing the same for `user.shares`.

Minor Issues

No minor issues have been identified.

Warnings

2. Non-Existent Test Coverage for DAO package

No `tests` for the DAO package have been provided. This means that it has not been possible for the auditors to assess testing. Certain bugs and vulnerabilities are very hard to detect during a code audit without test assessment.

Recommendation

Provide unit test coverage for the package.

Informative Notes

3. `Api3Template.sol`: Element count discrepancy for `_votingSettings` array

The documentation states that `_votingSettings` should be an array of 4 elements, while the actual parameter declaration declares it as an array of 3 elements only.

Recommendation

Resolve the array element count discrepancy between the code and documentation.

Disclaimer

Solidified audit is not a security warranty, investment advice, or an endorsement of API3 DAO or its products. This audit does not provide a security or correctness guarantee of the audited smart contract. Securing smart contracts is a multistep process, therefore running a bug bounty program as a complement to this audit is strongly recommended.

The individual audit reports are anonymized and combined during a debrief process, in order to provide an unbiased delivery and protect the auditors of Solidified platform from legal and financial liability.

Solidified Technologies Inc.