Super Protocol

Super Protocol Staking Security Analysis

by Pessimistic

This report is public

May 31, 2022

Abstract

In this report, we consider the security of smart contracts of Super Protocol Staking project. Our task is to find and describe security issues in the smart contracts of the platform.

Disclaimer

The audit does not give any warranties on the security of the code. A single audit cannot be considered enough. We always recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts. Besides, a security audit is not investment advice.

Summary

In this report, we considered the security of the Super Protocol Staking smart contracts. We performed our audit according to the procedure described below.

The audit showed one issue of medium severity: ERC20 standard violation.

Also, one low-severity issue was found. Our only concern regarding the code quality was put into the notes section.

After the initial audit, the codebase was updated. In this update, the developers fixed all the previously discovered issues. However, our concern in the notes section is still relevant.

The project also received an additional update, in which the developer team increased the quality of the codebase.

General recommendations

We recommend improving NatSpec coverage.

Project overview

Project description

For the audit, we were provided with the Super Protocol Staking project on a private GitHub repository, commit 4809c9e2e303c1d066150eb0bf2d6167b0ce3108.

The project has a documentation file with its description.

All 19 tests pass, test coverage is 100%.

The total LOC of audited sources is 130.

Codebase update #1

After the initial audit the codebase was updated. For the recheck we were provided with commit 2fde09f78e7357c164566b991d56dc972172ea77.

In this update, ERC20 standard violation and Project management issues were fixed.

Codebase update #2

For an additional recheck we were provided with commit 9820fc8f9f56aebc02309496bc413b08ca0089f6.

In this update, the developer team added changes related to the code quality of test scripts and smart contracts. The functionality of the project remained untouched.

Procedure

In our audit, we consider the following crucial features of the code:

1. Whether the code is secure.
2. Whether the code corresponds to the documentation (including whitepaper).
3. Whether the code meets best practices.

We perform our audit according to the following procedure:

- Automated analysis
  - We scan the project's codebase with the automated tools: Slither.
  - We manually verify (reject or confirm) all the issues found by the tool.
- Manual audit
  - We manually analyze the codebase for security vulnerabilities.
  - We assess the overall project structure and quality.
- Report
  - We reflect all the gathered information in the report.

Manual analysis

The contracts were completely manually analyzed, their logic was checked. Besides, the results of the automated analysis were manually verified. All the confirmed issues are described below.

Critical issues

Critical issues seriously endanger project security. They can lead to loss of funds or other catastrophic consequences. The contracts should not be deployed before these issues are fixed.

The audit showed no critical issues.

Medium severity issues

Medium issues can influence project operation in the current implementation. Bugs, loss of potential income, and other non-critical failures fall into this category, as well as potential problems related to incorrect system management. We highly recommend addressing them.

M01. ERC20 standard violation (fixed)

ERC-20 standard states: Callers MUST handle `false` from returns `bool success`. Callers MUST NOT assume that `false` is never returned.

However, returned values from `transfer` calls at lines 42, 129, 141, and `transferFrom` calls at line 108 are not checked.

The issue has been fixed and is not present in the latest version of the code.

Low severity issues

Low severity issues do not directly affect project operation. However, they might lead to various problems in future versions of the code. We recommend fixing them or explaining why the team has chosen a particular option.

L01. Project management (fixed)

OpenZeppelin `ERC20` and `Math` libraries are copied into the project repository. We recommend managing them as dependencies.

The issue has been fixed and is not present in the latest version of the code.

Notes

N01. Code quality

Start and finish of the staking process depend on block timestamps. Block timestamps might be unpredictable as they are dependent on the mining process.

This analysis was performed by Pessimistic:

Pavel Kondratenkov, Security Engineer
Nikita Kirillov, Junior Security Engineer
Irina Vikhareva, Project Manager
Alexander Seleznev, Founder

May 31, 2022