ANCORE Strategy

Fananees NFT Security Analysis

by Pessimistic

This report is public

April 21, 2022

Abstract

In this report, we consider the security of smart contracts of Fananees NFT project. Our task is to find and describe security issues in the smart contract 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 Fananees NFT smart contracts. We performed our audit according to the procedure described below.

The audit showed several issues of medium severity including a Bug, Overpowered Owner and Tests Issue.
Also, one low-severity issue was found. The overall code quality is good.

Our two concerns about functionality were put into notes section.

After the initial audit, the codebase was updated. In this update, the developers fixed all of the previously discovered issues except Overpowered Owner. In addition to this, the developers added new functionality that is not described in the documentation.

General recommendations

We recommend fixing Overpowered Owner issue and improving NatSpec coverage.

Project overview

Project description

For the audit, we were provided with Fananees NFT project on a private GitHub repository, commit 59b5aca5a8812d58bfb47988e51007a5cfaf791f.

The project has README.md file with a short description of the project. The code has some NatSpec comments. However, there is no detailed documentation available.

All 32 tests pass, the code coverage is 68,97%.

The total LOC of audited sources is 147.

Codebase update #1

After the initial audit, we were provided with commit c2c6982b667d63e355b14f7c4f9029de1fefd46d.

In this update, the developers fixed most of the issues and added new functionality. However, the documentation has not been updated and some aspects of new functionality cannot be checked (see Documentation issue). The number of passing tests has increased to 57, code coverage is 82,65%

The total LOC of audited sources is 291.

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 tool Slither.
  - We manually verify (reject or confirm) all the issues found by the tools.
- 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 only provided contract `FananeesNFT.sol` was completely manually analyzed, logic of the contract 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. Bug (fixed)

In `require` comparison of the same variable leads to a constant revert of the function in line 221.

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

M02. Overpowered Owner

- Owner can infinitely increase lock-time of tokens in lines 213-218.
- Owner can reduce the number of available tokens to mint for team and VC at lines 220-226.

The developers changed mentioned functionality. However, they implemented the following functions that we consider as overpowered: `changeTeamAndVCLimit`, `changeBinanceSaleLimit`, `changePrice`, `setWhitelistClaimsLimit`.

M03. Tests issue (fixed)

The project has tests. However, the overall code coverage is only 68.97%. Testing is crucial for the security of the project, and the audit does not replace tests in any way. We highly recommend covering the code with tests and ensuring that all tests pass and the code coverage is sufficient.

The issue has been fixed and is not present in the latest version of the code. After code update coverage has increased up to 82,65%

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. Gas consumption (fixed)

Consider declaring functions as external instead of public where possible to improve code readability and optimize gas consumption.

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

Notes

N01. Minting during Public stage may not be possible (fixed)

If during PreLaunch and Whitelist stages tokens got to the `MAX_SUPPLY` limit then minting during Public stage will be impossible.

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

N02. The limit of teamAndVC tokens may not be reached (fixed)

If during PreLaunch the `teamAndVCLimit` was not reached and during Whitelist and Public stages the `MAX_SUPPLY` limit was reached then the `teamAndVCLimit` will never be reached.

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

N03. Documentation issue (new)

- `BinanceSale` functionality has been added. This functionality is not described in `README.md` - the only provided documentation file.
- Variables `WHITELIST_LENGTH` and `whitelistClaimsLimit` were changed. However, this does not correlate with information in `README.md` file.

This analysis was performed by Pessimistic:

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

April 21, 2022