Swarm Markets Action Manager Security Analysis

by Pessimistic

This report is public

November 9, 2021

Abstract 2
Disclaimer 2
Summary 2
General recommendations 2
Project overview 3
Project description 3
Code base update 3
Procedure 4
Manual analysis 5
Critical issues 5
Medium severity issues 6
Overpowered role 6
Tests issues (fixed) 6
No documentation 6
Low severity issues 7
Code quality 7

Abstract

In this report, we consider the security of smarts contracts of Swarm Markets Action Manager 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. One 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, security audit is not an investment advice.

Summary

In this report, we considered the security of Swarm Markets Action Manager smart contracts. We performed our audit according to the procedure described belows.

The initial audit showed three issues of medium severity including Overpowered role and Tests issues. Also, a few issues of low severity were found.

The project has no documentation.

After the initial audit, the code base was updated. Several issues were fixed with this update.

General recommendations

We recommend adding the documentation and addressing Overpowered role issue. We also recommend adding CI to run tests, calculate code coverage, and analyze code with linters and security tools.

Project overview

Project description

For the audit, we were provided with Swarm Markets Action Manager project on a private GitHub repository, commit 4f07c79745b8d85d486a0e882aeabe2fcba061ec.

The scope of the audit included only **ActionManager.sol** file.

The project has no documentation. However, the code base has detailed NatSpec comments.

Two tests out of 352 do not pass, the code coverage calculation fails.

The total LOC of audited sources is 264.

Code base update

After the initial the audit, the code base was updated. For the recheck, we were provided commit 8be1ca473905e9c3bbece6e119a9826407b2f6f7.

In this update, several issues were fixed. Also, tests issues were fixed, the code coverage is 100%.

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 project's code base with automated tools: Slither and SmartCheck.
  - We manually verify (reject or confirm) all the issues found by tools.
- Manual audit
  - We manually analyze code base for security vulnerabilities.
  - We assess 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 often lead to the loss of funds or other catastrophic failures. The contracts should not be deployed before these issues are fixed. We highly recommend fixing them.

The audit showed no critical issues.

Medium severity issues

Medium issues can influence project operation in 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.

Overpowered role

The admin role can change the addresses of the contracts that the system depends on. As a result, the admin can manipulate the behavior of the system. E.g., the admin can stole users' tokens by changing `xTokenWrapperAddress` address.

In the current implementation, the system depends heavily on the admin role. Thus, there are scenarios that can lead to undesireable consequences to the project and its users, e.g., if the private keys of the admin become compromised.

We recommend designing contracts in a trustless manner or implementing proper key management, e.g., multisig.

Tests issues (fixed)

The project has tests. However, two tests out of 352 do not pass.

Testing is crucial for code security and audit does not replace tests in any way. We highly recommend covering the code with tests and making sure that all tests pass and the code coverage is sufficient.

All tests pass in the latest version of the code. The code coverage is 100%.

No documentation

The project has no documentation. Considering the importance of the ActionManager contract and its role in the project, the documentation is a critical part that helps to improve security and reduce risks.

Proper documentation should explicitly describe the purpose and behavior of the contracts, their interactions, and main design choices. It is also essential for any further integrations.

Low severity issues

Low severity issues do not directly affect project's operations. However, they might lead to various problems in the future versions of the code. We recommend taking them into account.

Code quality

- In `_checkSwapFee()` function, consider using `1e18` syntax instead of `10**18` when declaring `bone` variable at line 395.

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

- We recommend checking the returned values of external calls at lines 250 and 264.

  The issues have been fixed and are not present in the latest version of the code.

- Variable `bone` at line 395 is redundant since it is only used to assign the value of `maxSwapFee` variable. Also, the value of `maxSwapFee` is never changed. Therefore, consider declaring it as a `constant`.

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

- In `_deployXPoolToken()` function, consider declaring a `constant` instead of a literal to pass to `xTokenFactoryContract.deployXToken()` call at line 378.

This analysis was performed by Pessimistic:

Evgeny Marchenko, Senior Security Engineer
Daria Korepanova, Security Engineer
Boris Nikashin, Analyst
Irina Vikhareva, Project Manager

November 9, 2021