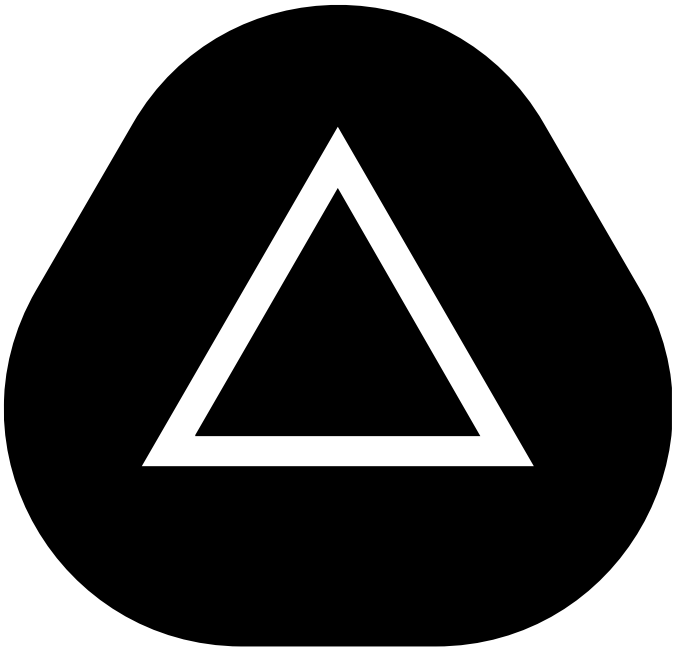

ARTM Security Analysis

by Pessimistic

This report is public.

Published: December 21, 2021

Abstract

In this report, we consider the security of smart contracts of ARTM 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 ARTM token smart contracts. We performed our audit according to the procedure described below.

The initial audit showed Timelock mapping content issue of medium severity and a few issues of low severity. The project had no documentation.

After the initial audit, the code base was updated. In this update, most of the issues were fixed, including Timelock mapping content issue. Also, the documentation was added to the project.

After the recheck, another update was performed. With this update, the project changed its name, the code base moved to a public repository.

General recommendations

We recommend implementing CI to run tests and analyze code with linters and security tools.

Project overview

Project description

For the audit, we were provided with Artemis project on a private GitHub repository, commit cd889703d9274906dda53aa9eccd2b257b487d54.

The project consists of two parts: **token/** and **crowdsale/** folders. Crowdsale part has no tests. For the Token part, some tests do not pass, the coverage is 39.13%.

The code compiles without any issues, but optimization is turned off.

The project had no documentation. However, the code has detailed NatSpecs.

The total LOC of audited sources is 183.

Code base update #1

After the initial audit, the code base was updated. For the recheck #1, we were provided with commit eafb99d2144ae118b8ea30803b8231ad419e622d.

The documentation was added to the project: **Artemis (ARTM) Smart Contracts Overview.pdf** file, sha1sum 15a3445be7067738be1f747f9c81d0304965fd99.

Also, new tests were added, tests that did not pass were fixed, the coverage was improved up to 100%.

Code base update #2

After the recheck #1, the code base moved to ARTM public GitHub repository. For the recheck #2, we were provided with commit 7b8ad2415199ffcdac0709f8f50d4ac2362c56fb. In this update, only the name of the project was changed.

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 tool 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 smart contracts security. We highly recommend fixing them.

The audit showed no critical issues.

Medium severity issues

Medium issues can influence project operation in current implementation. We highly recommend addressing them.

`TimeLocks` mapping content (fixed)

`createTimeLock()` function of **ArtemisTimeLockFactory** contract can be accessed by anyone. The function has no proper checks if a timelock with provided `_timeLockId` already exists. As a result, a random person can not only add arbitrary records to `timeLocks` mapping but also replace existing timelocks, messing up its content.

Since there is no documentation provided, it is hard to tell how this can affect the project.

We recommend adding the check whether the timelock with provided `_timeLockId` has been added already or limiting access to `createTimeLock()` function.

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

No documentation (fixed)

The project has no documentation. Considering the importance of the Token and Crowdsale smart contract and their 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.

The documentation was added to the project.

Tests issues (fixed)

The **ArtmCrowdsale** contract is not covered with tests. For the Token part of the project, some tests do not pass. The overall coverage is below 40%.

Testing is crucial for code security and audit does not replace tests in any way.

We highly recommend both covering the code with tests and making sure that the test coverage is sufficient.

New tests were added in the latest version of the code. All tests pass, the coverage is 100%.

Low severity issues

Low severity issues can influence project operation in future versions of code. We recommend taking them into account.

Code quality (fixed)

- In **ArtemisTimeLock** contract, providing initial `0` value for `unit256` variable is unnecessary at lines 29 and 37.
- In **ArtemisTimeLock** contract, consider declaring `release()` function at line 107 as `external` instead of `public`.

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

Dependency management

- **package-lock.json** file is not included into repository. Moreover, it is added to **.gitignore** file.

  The issue has been fixed and is not present in the latest version of the code.
- Scripts in **package.json** require `mocha` to be installed globally though it is in `devDependencies`.

Gas consumption

The compiler optimization is turned off.

This analysis was performed by Pessimistic:

Evgeny Marchenko, Senior Security Engineer
Igor Sobolev, Security Engineer
Daria Korepanova, Security Engineer
Boris Nikashin, Analyst
Alexander Seleznev, Founder

December 21, 2021