Beam Swaps Security Analysis

by Pessimistic

This report is public.

Published: February 6, 2021

Abstract

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

The initial audit showed three project-level issues (no tests, no configuration files, and incomplete documentation) and several issues of low severity (mostly, code quality issues).

After the audit, the code base was updated to the latest version. Tests and configuration files were added to the repository, most of the mentioned issues were fixed.

General recommendations

We recommend completing the documentation to the project.

Procedure

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

1. Whether code logic corresponds to the specification.
2. Whether the code is secure.
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 compare the project's code to Compound's codebase and check whether the changes are consistent with documentation.
  - We manually analyze new code for security vulnerabilities.
  - We assess overall project structure and quality.
- Report
  - We reflect all the gathered information in the report.

Project overview

Project description

In our analysis we consider smart contracts of Beam project on GitHub repository, commit 3ea739cebca5e53c2db41943a8f1f3ee6451c19a.

The documentation for the project only includes BTC-favored atomic swaps description. It was provided on GitHub wiki, commit dfce969082c868bc59efa53254e4bbcef766c34d.

The project has no tests and no configuration files.

The total LOC of audited sources is 372.

Latest version of the code

After the initial audit, the code base was updated. For the recheck, we were provided with commit 515e91b2a8a8c8403c32b169e715b42f8c32099d.

Tests and configuration files were added to the project.

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.

No tests (fixed)

The provided code does not contain tests. 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.

No configuration files (fixed)

The project has no configuration files. The configuration files are essential for running tests, deploying contracts, and integrating/using additional tools like tests coverage and linters.

Low severity issues

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

Code quality

- In the code, pragma solidity declaration includes two major solidity versions:

  ```
  pragma solidity >=0.6.0 <0.8.0;
  ```

  We recommend specifying the version of the compiler explicitly, including minor version, i.e. `pragma solidity 0.7.6;`.

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

- Modifiers are not reused in the contracts, their logic is detached from the functions, which makes the code harder to read. Therefore, we recommend logic of these modifiers to corresponding functions to improve code readability.

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

- There is a misleading name used in **swap_contract_aggregate_signature.sol** and **erc20_swap_contract_aggregate_signature.sol** files for an address parameter — `address hashedSecret`. We recommend using more clarifying names.

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

- `msg.sender` check is redundant in `isRefundable` and `isRedeemable` modifiers in all contracts as the recipient is fixed in each contract.

Signature malleability (fixed)

EIP-2 allows signature malleability for `ecrecover()` opcode.

`redeem()` function uses `ecrecover()` in **swap_contract_aggregate_signature.sol** and **erc20_swap_contract_aggregate_signature.sol** files to verify signature.

We recommend implementing additional checks of users' signatures. See ECDSA library from OpenZeppelin for more details.

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

Possible out-of-gas (fixed)

In **swap_contract.sol** and **swap_contract_aggregate_signature.sol** files, `transfer()` function has a limit of 2300 gas for the called contract, which might be insufficient for receiver's fallback function. In such cases, it is recommended to send ether via `.call()` method as it has lower chances to run out of gas. In the worst scenario, one of the parties will be unable to receive their ether.

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

Incomplete documentation

The provided documentation only has BTC-favored atomic swaps description. The description for other swaps is missing.

We recommend adding the documentation for the rest of the project.

This analysis was performed by Pessimistic:

Evgeny Marchenko, Senior Security Engineer
Boris Nikashin, Analyst
Alexander Seleznev, Founder

February 6, 2021