Morpho Vaults Security Analysis

by Pessimistic

This report is public

February 20, 2023

Abstract

In this report, we consider the security of smart contracts of Morpho Vaults 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 Morpho Vaults smart contracts. We described the audit process in the section below.

Morpho Vaults deeply integrate with the Morpho protocol itself, duplicating risks of using it.

The audit showed only several issues of low severity. They do not endanger project security.

After the initial audit, the codebase was updated.

The developers fixed all the low severity issues. And we removed L01 as a false positive issue.

General recommendations

We recommend improving the code coverage of tests.

Project overview

Project description

For the audit, we were provided with Morpho Vaults project on a public GitHub repository, commit 43501d887d8e9e1bae832d5269ee0420483b395e.

The scope of the audit includes **src/ERC4626UpgradeableSafe.sol** file and contracts inside **src/compound** and **src/aave-v2** folders.

The documentation for the project includes a page on a public website: https://developers.morpho.xyz/interact-with-morpho/erc-4626-vaults and a private Yellow Paper (sha1sum aea99e209625315802f15a347fb0b2114be3987d). Also, the developers provided the code with detailed NatSpec comments.

All 57 tests pass successfully. The code coverage is 74.6%.

The total LOC of audited sources is 416.

Codebase update

After the initial audit, the codebase was updated. For the recheck, we were provided with commit b06fb3fab3f0bfa1b104a285fc13eb61a228e704.

The scope of the audit consisted of changes to the previous scope.

This update included fixes for all the low severity issues. The number of tests did not change. The issue L01 was removed as false positive.

Audit process

We started the audit on February 6, 2023 and finished on February 14, 2023.

We inspected the materials provided for the audit. Then, we contacted the developers for an introduction to the project. After a discussion, we performed preliminary research and specified those parts of the code and logic that require additional attention during an audit:

- The ERC-4626 standard is implemented correctly.
- Contracts contain functionality for claiming Compound supply rewards.
- Morpho supply functionality is integrated correctly.
- The reward mechanism cannot be manipulated via flashloans.

During the work, we stayed in touch with the developers and discussed confusing or suspicious parts of the code.

We manually analyzed all the contracts within the scope of the audit and checked their logic. Among other, we verified the following properties of the contracts:

- Gas usage is optimized.
- Upgradeability patterns are implemented correctly.
- Differences between underlying tokens cannot disrupt the vaults.
- Compound vault users can always claim their comp rewards, and the balance will always be sufficient.
- Math libraries are used correctly.

We scanned the project with Slither analyzer and manually verified all the occurrences found by it.

We ran tests and calculated the code coverage.

We combined in a private report all the verified issues we found during the manual audit or discovered by automated tools.

After our audit, the developers immediately fixed all the low severity issues. In addition, we removed L01 as a false positive since the developers explained to us why it was not the issue.

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 severity 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.

The audit showed no issues of medium severity.

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.

L02. Redundant using for (fixed)

The `RayWadMath` and `CompoundMath` libraries are not used in **SupplyVaultBase** contracts, so the corresponding `using for` expressions are redundant.

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

L03. Unused return value (fixed)

`__SupplyVaultBase_init()` function returns a `isEth` boolean flag, which is ignored. Consider removing it in `initialize()` function of **compound/SupplyVault** contract.

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

Notes

N01. No slippage protection

The standard functions - `deposit, mint, withdraw, redeem` - are susceptible to slippage because the changes in liquidity affect the number of shares that will be minted or burned for a single unit of asset. Although it is impossible to set up a typical slippage protection (i.e., `minShares` parameter in a `deposit` function) and preserve an ERC-4626 standard simultaneously, this could become an issue for vault users.

N02. Possible reaching the approve limit

The vaults give Morpho infinite approval as `type(uint256).max`. Some underlying tokens, which inherit an ERC20 implementation of OpenZeppelin under the 4.4 version, actually decrease allowance even if it is infinite. Every time a user supplies such an asset to a Morpho vault, its allowance decreases.

Although it is unlikely that a 2^256 allowance will ever go down to zero, this issue is worth noting as the vault is not able to approve more.

N03. Morpho vault effect on the P2P queue

Morpho sees a tokenized vault as its individual supplier, but because the vault collects liquidity from all its users, it can easily lead the P2P queue as the biggest supplier. In this case, the vault will be the first candidate to go P2P, while smaller Morpho suppliers will be pushed down the queue.

N04. Lack of pools status check

Aave-v2 and Compound pools can be paused or inactive, which makes it impossible to supply to them. Consider adding corresponding checks, i.e. `LendingPool.paused()` in case of Aave-V2.

This analysis was performed by Pessimistic:

Evgeny Marchenko, Senior Security Engineer
Daria Korepanova, Security Engineer
Vladimir Pomogalov, Security Engineer
Ivan Gladkikh, Security Engineer
Yhtyyar Sahatov, Junior Security Engineer
Irina Vikhareva, Project Manager

February 20, 2023