PUBLIC

Code Assessment

of the Yearn ERC4626 Router Smart Contracts

August 29, 2023

Produced for

by

Contents

1 Executive Summary

Dear Yearn team,

Thank you for trusting us to help Yearn with this security audit. Our executive summary provides an overview of subjects covered in our audit of the latest reviewed contracts of Yearn ERC4626 Router according to Scope to support you in forming an opinion on their security risks.

Yearn implements a router contract to migrate, deposit and withdraw from various vaults. Additionally, it simplifies some of the user actions like wrapping ether and providing the possibility to perform multi-calls.

We did not uncover any severe issues. In summary, we find that the codebase provides a high level of security.

It is important to note that security audits are time-boxed and cannot uncover all vulnerabilities. They complement but don't replace other vital measures to secure a project.

The following sections will give an overview of the system, our methodology, the issues uncovered and how they have been addressed. We are happy to receive questions and feedback to improve our service.

Sincerely yours,

ChainSecurity

1.1 Overview of the Findings

Below we provide a brief numerical overview of the findings and how they have been addressed.

Critical-Severity Findings	0
High-Severity Findings	0
Medium-Severity Findings	0
Low-Severity Findings	0

2 Assessment Overview

In this section, we briefly describe the overall structure and scope of the engagement, including the code commit which is referenced throughout this report.

2.1 Scope

The assessment was performed on the source code files inside the Yearn ERC4626 Router repository based on the documentation files. The table below indicates the code versions relevant to this report and when they were received.

V	Date	Commit Hash	Note
1	14 July 2023	68a9f4c354063372931be093cb4587357981f2d6	Initial Version

For the solidity smart contracts, the compiler version `0.8.18` was chosen. However, the inherited external contracts have different compiler versions:

- `Multicall` has a foating pragma `>=0.7.6`
- `PeripheryPayments` has a foating pragma `>=0.7.5`
- `SelfPermit` has a foating pragma `>=0.5.0`

All solidity files in the repository's `src` folder were in scope except for test and mock files.

2.1.1 Excluded from scope

All test and mock files are excluded from the scope.

2.2 System Overview

This system overview describes the initially received version (Version 1) of the contracts as defined in the Assessment Overview.

Furthermore, in the findings section, we have added a version icon to each of the findings to increase the readability of the report.

Yearn offers a router to improve usability by offering multi-call functionality to e.g., deposit, withdraw or migrate to other yearn vaults.

The main contract is the Router contract which has the following main functionalities:

- Minting/depositing funds into the vaults: `mint`, `deposit`, `depositToVault` (different variations)
- Withdraw/redeeming funds from the vaults: `withdraw`, `withdrawDefault`, `redeem` (different variations), `redeemDefault`. `withdrawDefault` and `redeemDefault` use the default ERC4626 corresponding functions while `withdraw` and `redeem` use Yearn-specific functions
- Migrating funds from one vault to another: `migrate` (different variations), `migrateFromV2` (different variations). `migrateFromV2` is used to migrate from a Vault V2 that requires the caller to hold the vault's shares.
- Making multiple calls to the contract in one transaction: `multicall`
- Transferring ERC20 tokens from the message sender to the contract using `transferFrom`: `pullToken`

- Sending ERC20 tokens from the contract to some address: `sweepToken`
- Giving an ERC20 approval to a spender for the contract: `approve`
- Various ERC20 permit related functions: , `selfPermit`, `selfPermitIfNecessary`, `selfPermitAllowed` and `selfPermitAllowedIfNecessary`
- sending the balance of the contract to the message sender: `refundETH`
- Unwrapping and Wrapping `ETH`: `wrapWETH9` and `unwrapWETH9`

Note that the functions `depositToVault`, `redeem`, `migrate` and `migrateFromV2` are overloaded multiple times with different default arguments.

3 Limitations and use of report

Security assessments cannot uncover all existing vulnerabilities; even an assessment in which no vulnerabilities are found is not a guarantee of a secure system. However, code assessments enable the discovery of vulnerabilities that were overlooked during development and areas where additional security measures are necessary. In most cases, applications are either fully protected against a certain type of attack, or they are completely unprotected against it. Some of the issues may affect the entire application, while some lack protection only in certain areas. This is why we carry out a source code assessment aimed at determining all locations that need to be fixed. Within the customer-determined time frame, ChainSecurity has performed an assessment in order to discover as many vulnerabilities as possible.

The focus of our assessment was limited to the code parts defined in the engagement letter. We assessed whether the project follows the provided specifications. These assessments are based on the provided threat model and trust assumptions. We draw attention to the fact that due to inherent limitations in any software development process and software product, an inherent risk exists that even major failures or malfunctions can remain undetected. Further uncertainties exist in any software product or application used during the development, which itself cannot be free from any error or failures. These preconditions can have an impact on the system's code and/or functions and/or operation. We did not assess the underlying third-party infrastructure which adds further inherent risks as we rely on the correct execution of the included third-party technology stack itself. Report readers should also take into account that over the life cycle of any software, changes to the product itself or to the environment in which it is operated can have an impact leading to operational behaviors other than those initially determined in the business specification.

4 Terminology

For the purpose of this assessment, we adopt the following terminology. To classify the severity of our findings, we determine the likelihood and impact (according to the CVSS risk rating methodology).

- *Likelihood* represents the likelihood of a finding to be triggered or exploited in practice
- *Impact* specifies the technical and business-related consequences of a finding
- *Severity* is derived based on the likelihood and the impact

We categorize the findings into four distinct categories, depending on their severity. These severities are derived from the likelihood and the impact using the following table, following a standard risk assessment procedure.

Likelihood	Impact		
	High	Medium	Low
High	Critical	High	Medium
Medium	High	Medium	Low
Low	Medium	Low	Low

As seen in the table above, findings that have both a high likelihood and a high impact are classified as critical. Intuitively, such findings are likely to be triggered and cause significant disruption. Overall, the severity correlates with the associated risk. However, every finding's risk should always be closely checked, regardless of severity.

5 Findings

In this section, we describe our findings. The findings are split into these different categories:

Below we provide a numerical overview of the identified findings, split up by their severity.

Critical-Severity Findings	0
High-Severity Findings	0
Medium-Severity Findings	0
Low-Severity Findings	0

6 Notes

We leverage this section to highlight further findings that are not necessarily issues. The mentioned topics serve to clarify or support the report, but do not require an immediate modification inside the project. Instead, they should raise awareness in order to improve the overall understanding.

6.1 Use of Atomic Transactions Only

Note Version 1

The router performs critical actions, of which many should not be done separately in multiple transactions. All critical operations must be done in one atomic transaction.