PUBLIC

Code Assessment

of the Partial Migration to V3 Smart Contracts

October 18, 2023

Produced for

by

Contents

1 Executive Summary

Dear Gearbox team,

Thank you for trusting us to help Gearbox Protocol with this security audit. Our executive summary provides an overview of subjects covered in our audit of the latest reviewed contracts of Partial Migration to V3 according to Scope to support you in forming an opinion on their security risks.

Gearbox Protocol implements a new interest rate model for the lending pools and an updated address provider contract. The goal of this report is to assess the security of these contracts as they are intended to be used with the older versions of the Gearbox system. This is a partial report of a more extensive in-progress security assessment for Gearbox V3.

The most critical subjects covered in our audit are functional correctness, access control, and backward compatibility. Security regarding all the aforementioned subjects is high.

The general subjects covered are code complexity, precision of arithmetic operations, and specification. Security regarding all the aforementioned subjects is high.

In summary, we find that the codebase provides a high level of security.

As this is a partial report, it is important to note that the assessment is ongoing, and the findings and recommendations provided are subject to change in future reports which include the contracts in scope.

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	1
• Code Corrected	1

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 Partial Migration to V3 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	06 June 2023	527b365c97abcface94da4872c4c05a3f1e70d6b	Initial Version
2	18 Oct 2023	519647cc73f74db3af3730549e450e19e994d0d8	Updated Version

For the solidity smart contracts, the compiler version `0.8.17` was chosen.

The following contracts are included in scope:

```
core:
    AddressProviderV3.sol

pool:
    LinearInterestRateModelV3.sol
```

Regarding the backward compatibility, the new contracts were checked to implement all the getter functions which are present in the contracts to be replaced.

2.1.1 Excluded from scope

All the contracts not explicitly in scope are considered out-of-scope.

2.2 System Overview

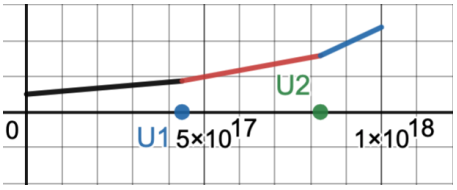
This system overview describes the initially received version (Version 1) of the contracts as defined in the Assessment Overview.

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

Gearbox Protocol offers a new base interest rate model for the liquidity pools, and an updated version of the `AddressProvider`.

2.3 2-Points Linear Model

The linear model for the base interest rate of the Pools is updated to be a 2-points linear model like the one drawn below:



New pools can optionally revert during borrowing if the borrow rate goes beyond the second inflexion point. Old pools do not have this option and thus will always be able to push utilization beyond `U2` when borrowing.

2.4 AddressProviderV3

The new `AddressProvider` introduces versioning, the addresses are now stored behind a `(key, version)` mapping. The following functions and their mapping are available for backward compatibility with Gearbox protocol V2:

- `getACL()`: `("ACL", 0)`
- `getContractsRegister()`: `("CONTRACTS_REGISTER", 0)`
- `getPriceOracle()`: `("PRICE_ORACLE", 2)`
- `getAccountFactory()`: `("ACCOUNT_FACTORY", 0)`
- `getDataCompressor()`: `("DATA_COMPRESSOR", 2)`
- `getTreasuryContract()`: `("TREASURY", 0)`, note that the previous key in V2 was `"TREASURY_CONTRACT"`.
- `getGearToken()`: `("GEAR_TOKEN", 0)`
- `getWethToken()`: `("WETH_TOKEN", 0)`
- `getWETHGateway()`: `("WETH_GATEWAY", 1)`
- `getLeveragedActions()`: `("ROUTER", 1)`, note that the previous key in V2 was `"LEVERAGED_ACTIONS"`.

2.5 Trust Model

- The deployer of `LinearInterestRateModel` is trusted to set the parameters correctly and in a non-adversarial manner.
- The `configurator` of `AddressProviderV3` is assumed to be the governance `Timelock` contract and is trusted to set the addresses correctly and in a non-adversarial manner. The `configurator` is also trusted to set the addresses for backward compatibility as listed above.

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 any open findings. Findings that have been resolved have been moved to the Resolved Findings section. The findings are split into these different categories:

- Design: Architectural shortcomings and design inefficiencies

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 Resolved Findings

Here, we list findings that have been resolved during the course of the engagement. Their categories are explained in the Findings section.

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	1

- Division By Zero Code Corrected

6.1 Division By Zero

Design Low Version 1 Code Corrected

CS-GEARV3PREMIGR-002

`LinearInterestRateModelV3.availableToBorrow()` calculates the `U_WAD` value by dividing with `expectedLiquidity`. However, the corner case exists for an empty pool, where the `expectedLiquidity` is 0.

Code corrected:

A check was added to take care of the case where `expectedLiquidity` is 0.

7 Informational

We utilize this section to point out informational findings that are less severe than issues. These informational issues allow us to point out more theoretical findings. Their explanation hopefully improves the overall understanding of the project's security. Furthermore, we point out findings which are unrelated to security.

7.1 Missing Input Sanitization

Informational **Version 1**

CS-GEARV3PREMIGR-001

The constructor of `AddressProviderV3` take the address `_acl` as parameter, but the address is not checked to be non-zero. There is no security issue as the contract would simply be unusable.