Huma contracts v2 PR 4812

Security Review

Review by:
0xLeastwood, Lead Security Researcher

November 22, 2024

Contents

1 Introduction

1.1 Disclaimer

A security review a detailed evaluation of the security posture of the code at a particular moment based on the information available at the time of the review. While the review endeavors to identify and disclose all potential security issues, it cannot guarantee that every vulnerability will be detected or that the code will be entirely secure against all possible attacks. The assessment is conducted based on the specific commit and version of the code provided. Any subsequent modifications to the code may introduce new vulnerabilities that were absent during the initial review. Therefore, any changes made to the code require a new security review to ensure that the code remains secure. Please be advised that a security review is not a replacement for continuous security measures such as penetration testing, vulnerability scanning, and regular code reviews.

1.2 Risk assessment

Severity	**Description**
Critical	*Must* fix as soon as possible (if already deployed).
High	Leads to a loss of a significant portion (>10%) of assets in the protocol, or significant harm to a majority of users.
Medium	Global losses <10% or losses to only a subset of users, but still unacceptable.
Low	Losses will be annoying but bearable. Applies to things like griefing attacks that can be easily repaired or even gas inefficiencies.
Gas Optimization	Suggestions around gas saving practices.
Informational	Suggestions around best practices or readability.

1.2.1 Severity Classification

The severity of security issues found during the security review is categorized based on the above table. Critical findings have a high likelihood of being exploited and must be addressed immediately. High findings are almost certain to occur, easy to perform, or not easy but highly incentivized thus must be fixed as soon as possible.

Medium findings are conditionally possible or incentivized but are still relatively likely to occur and should be addressed. Low findings a rare combination of circumstances to exploit, or offer little to no incentive to exploit but are recommended to be addressed.

Lastly, some findings might represent objective improvements that should be addressed but do not impact the project's overall security (Gas and Informational findings).

2 Security Review Summary

Huma V2 is an on-chain lending protocol targeting business use cases. It is designed to run on EVM-compatible chains. The smart contracts are designed to be adaptive by supporting plugins for tranche policies, fee managers, and due managers

From Nov 7th to Nov 8th the security researchers conducted a review of huma-contracts-v2 on commit hash PR 4812.

The researchers reviewed Huma's contracts-v2 changes holistically on commit hashes 4c289261 (on develop) d01d248c and (on main) and determined that all issues were resolved and no new issues were identified.

A total of **1** issues in the following risk categories were identified:

- Critical Risk: 0
- High Risk: 0
- Medium Risk: 0
- Low Risk: 0
- Gas Optimizations: 0
- Informational: 1

3 Findings

3.1 Informational

3.1.1 Design considerations with the auto redemption feature

Severity: Informational

Context: Global scope

Description: Huma intends to make a small upgrade where the sentinel service account can submit redemption requests on behalf of the lender. A few of their design considerations made include:

- The default state for `autoRedemptionAfterLockup` in the pool config is `false`.
- The sentinel service account does not need to be approved by the lender.
- Once a redemption request has been initiated, if `autoRedemptionAfterLockup` is enabled, `cancelRedemptionRequest()` should revert.
- An autotask will add redemption requests for lenders after their lockup period expires.

The last point brings attention to Huma's definition of when a lockup is considered expired. For example, if the user wishes to keep their deposit locked for longer, they will need to re-deposit and reset their `depositRecord.lastDepositTime` to `block.timestamp`. This means if they don't perfectly time their re-deposit, they risk adding an additional `withdrawalLockoutPeriodInDays` to the lockup before the sentinel service account is able to auto-process the redemption.

Recommendation: Consider allowing users to define a lockup period that is greater than `withdrawalLockoutPeriodInDays` that is only configurable when there is no deposit record principal amount.

Huma: Thank you for pointing this out. The current behavior meets our requirements. For pools with the auto-redeem feature, we aim for all lenders to deposit within the first pay period only. This ensures that the pool matures no later than one pay period after the initial lock-up period.

For example, if a pool is activated on January 1 with a monthly pay period and a 90-day lock-up, we expect all deposits to occur in January, resulting in the pool maturing on May 1. We may also set the pool cap to zero at the end of the first pay period to prevent further deposits. However, we prefer not to hard-code this behavior into the contract to maintain flexibility as needed.

0xLeastwood: Acknowledged.