yAudit

November 24, 2025

Prepared for
Resupply

Audited by
HHK
adriro

Resupply CurveLend Operator

Smart Contract Security Assessment

Contents

1 Review Summary

1.1 Protocol Overview

The CurveLend Operator is a factory model for lending stables (crvUSD) into Curve lending markets (ERC-4626 vaults).

1.2 Audit Scope

This audit covers 2 smart contracts totaling approximately 120 lines of code across 1.5 days of review.

```
src/dao/
├── CurveLendMinterFactory.sol
└── CurveLendOperator.sol
```

1.3 Risk Assessment Framework

1.3.1 Severity Classification

Severity	Description	Potential Impact
Critical	Immediate threat to user funds or protocol integrity	Direct loss of funds, protocol compromise
High	Significant security risk requiring urgent attention	Potential fund loss, major functionality disruption
Medium	Important issue that should be addressed	Limited fund risk, functionality concerns
Low	Minor issue with minimal impact	Best practice violations, minor inefficiencies
Undetermined	Findings whose impact could not be fully assessed within the time constraints of the engagement. These issues may range from low to critical severity, and although their exact consequences remain uncertain, they present a sufficient potential risk to warrant attention and remediation.	Varies based on actual severity
Gas	Findings that can improve the gas efficiency of the contracts.	Reduced transaction costs
Informational	Code quality and best practice recommendations	Improved maintainability and readability

Table 1: severity classification

1.4 Key Findings

Breakdown of Finding Impacts

Impact Level	Count
🟥 Critical	0
🟧 High	0
🟨 Medium	0
🟩 Low	1
⬛ Informational	2

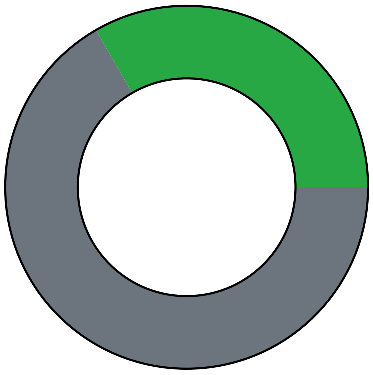

Figure 1: Distribution of security findings by impact level

1.5 Overall Assessment

The protocol presents a straightforward design for managing crvUSD allocation in a planned and permissioned fashion through its factory and operator architecture.

2 Audit Overview

2.1 Project Information

Protocol Name: Resupply
Repository: https://github.com/resupplyfi/resupply
Commit Hashes:

- 650fe16b9b45abd56e2f064da19cebdd2c9a9c0b
- 968bb07dc2e17fa30c7687c162460f872151fa2a

2.2 Audit Team

HHK, adriro

2.3 Audit Resources

Code repositories and documentation

2.4 Critical Findings

None.

2.5 High Findings

None.

2.6 Medium Findings

None.

2.7 Low Findings

2.7.1 Limit reduction amount

Technical Details

The `reduceAmount()` function can be used to withdraw an arbitrary amount of assets from the market permissionlessly. The implementation refunds the factory only the difference with respect to the limit, and then returns any leftovers to the vault.
This withdrawal and redeposit cycle could have a negative impact on the operator's assets if the vault implements entry and/or exit fees, or due to rounding issues during share conversion, depending on the attacker's capacity to inflate share value.

Impact

Low.

Recommendation

Validate the reduction amount is within the intended difference (i.e., `amount <= mintedAmount - mintLimit`).

Developer Response

Fixed in commit 002d863414af4c63f9b48e58d196049af91099dc.

2.8 Gas Savings Findings

None.

2.9 Informational Findings

2.9.1 Add a `shutdown` option when removing a market operator

Technical Details

When removing or replacing a market operator through `addMarketOperator()` or `removeMarketOperator()` the operator then needs to be called manually, once to reset the mint limit back to 0 through `setMintLimit()`, then to reimburse borrowed funds through `reduceAmount()` and finally `withdraw_profit()` to remove profits.
While this could be batched through a script and the Gnosis Safe interface, it might be easier and less prone to errors to update the functions directly.
To simplify DAO operations during operator removal/replacement, consider adding a `bool shutdown` parameter to `addMarketOperator()` and `removeMarketOperator()`, if set to `true` then call `setMintLimit(0)`, `reduceAmount(mintedAmount())` and `withdraw_profit()` on the operator.
Inside the operator contract, consider allowing `setMintLimit()` to be callable by the factory, not just the owner.

Impact

Informational.

Recommendation

Implement the suggested changes.

Developer Response

Acknowledged.

2.9.2 Consider using `previewRedeem()` to measure the operator's equity

The implementation utilizes `convertToAssets()` to calculate the total assets held, which may not accurately reflect the actual effects of a share redemption.

Technical Details

CurveLendMinterFactory.sol#L24. CurveLendOperator#L110.

Impact

Informational.

Recommendation

Change `convertToAssets()` for `previewRedeem()`.

Developer Response

Fixed in commit 002d863414af4c63f9b48e58d196049af91099dc.

2.10 Final Remarks

No significant issues were detected during the review. Auditors recommend exercising caution when interacting with vaults susceptible to share price manipulation, to avoid sandwich attacks or precision issues while deploying capital.