Coinbase MorphoWETHLoan-Policy Review 2

Security Review

Cantina Managed review by:
Slowfi, Lead Security Researcher
Akshay Srivastav, Security Researcher

April 16, 2026

Contents

1 Introduction **2**
- 1.1 About Cantina . . . 2
- 1.2 Disclaimer . . . 2
- 1.3 Risk assessment . . . 2
  - 1.3.1 Severity Classification . . . 2

2 Security Review Summary **3**
- 2.1 Scope . . . 3

3 Findings **4**
- 3.1 Informational . . . 4
  - 3.1.1 Duplicate market installation across Morpho protection variants . . . 4
  - 3.1.2 Deploy Function NatSpec Omits `MAX_TRIGGER_LTV_RATIO` environment variable . . . 4
  - 3.1.3 Redundant lowercase `WETH` view function duplicates public getter . . . 4
  - 3.1.4 Missing share slippage protection on Morpho vault deposits . . . 5
  - 3.1.5 A top up may not bring user's ltv below `triggerLtv` threshold . . . 5

1 Introduction

1.1 About Cantina

Cantina is a security services marketplace that connects top security researchers and solutions with clients. Learn more at cantina.xyz

1.2 Disclaimer

Cantina Managed provides 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 Cantina Managed 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 the Cantina Managed security review is not a replacement for continuous security measures such as penetration testing, vulnerability scanning, and regular code reviews.

1.3 Risk assessment

Severity level	Impact: High	Impact: Medium	Impact: Low
Likelihood: high	Critical	High	Medium
Likelihood: medium	High	Medium	Low
Likelihood: low	Medium	Low	Low

1.3.1 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 are 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

Base is a secure, low-cost, builder-friendly Ethereum L2 built to bring the next billion users onchain.

From Apr 8th to Apr 10th the Cantina team conducted a review of account-policies on commit hash 48870835. The team identified a total of **5** issues:

Issues Found

Severity	Count	Fixed	Acknowledged
Critical Risk	0	0	0
High Risk	0	0	0
Medium Risk	0	0	0
Low Risk	0	0	0
Gas Optimizations	0	0	0
Informational	5	0	5
Total	**5**	**0**	**5**

2.1 Scope

The security review had the following components in scope for account-policies on commit hash 48870835:

```
src/policies/MorphoWethLoanProtectionPolicy.sol
```

3 Findings

3.1 Informational

3.1.1 Duplicate market installation across Morpho protection variants

Severity: Informational

Context: MorphoLoanProtectionPolicy.sol#L70, MorphoLoanProtectionPolicy.sol#L212-L215, MorphoLoanProtectionPolicy.sol#L215, MorphoWethLoanProtectionPolicy.sol#L87-L98

Description: The function `_onSingleExecutorInstall` from contract `MorphoWethLoanProtectionPolicy` relies on the inherited `activePolicyByMarket` mapping to enforce the single active policy constraint for a market. Because that mapping is stored per policy contract instance, the same `(account, marketId)` can still be installed in `MorphoLoanProtectionPolicy` and `MorphoWethLoanProtectionPolicy` at the same time for a WETH collateral market. As a result, the account can hold two independent one-shot top-up authorizations for the same Morpho position, and the effective maximum authorized top-up for that market exceeds the bound implied by a single active protection policy.

Recommendation: Consider enforcing the `(account, marketId)` uniqueness check in shared storage across both Morpho protection variants, or consider making the ERC20 policy reject WETH collateral markets so only one protection contract can be active for a given Morpho market.

Coinbase: Acknowledged. No code fixes. Will discuss with product teams to avoid duplicate WETH policies in offchain implementations.

Cantina Managed: Acknowledged by the Coinbase team.

3.1.2 Deploy Function NatSpec Omits `MAX_TRIGGER_LTV_RATIO` environment variable

Severity: Informational

Context: Deploy.s.sol#L41

Description: The function `deploy` from contract `Deploy` reads three environment variables at runtime: `ADMIN`, `WETH`, and `MAX_TRIGGER_LTV_RATIO`. The `@dev` NatSpec comment on the function states that it reads only `ADMIN` and `WETH` from the environment, omitting any mention of `MAX_TRIGGER_LTV_RATIO`. The documentation is therefore inconsistent with the implementation, which may lead operators or integrators to believe the function has fewer external dependencies than it actually does.

Recommendation: Consider updating the `@dev` NatSpec on `deploy` to list all three environment variables it reads, and to note the valid range for `MAX_TRIGGER_LTV_RATIO` (greater than zero and strictly less than `1e18`) so operators understand the accepted input bounds before running the script.

Coinbase: Acknowledged. Will not fix as part of the audit.

Cantina Managed: Acknowledged by the Coinbase team.

3.1.3 Redundant lowercase `WETH` view function duplicates public getter

Severity: Informational

Context: MorphoWethLoanProtectionPolicy.sol#L75-L77

Description: The function `weth` from contract `MorphoWethLoanProtectionPolicy` is an external view function that returns the value of the public immutable state variable `WETH`. The Solidity compiler automatically generates an external getter for every public state variable using the variable's declared name. Because `WETH` is declared as `public`, the compiler already exposes an auto generated getter named `WETH()`. The manually declared `weth()` function is therefore a redundant alias that returns the same value through a separate selector without adding any behavior. The same pattern exists in the parent contract for the `morpho()` function and the `MORPHO` immutable.

Recommendation: Consider removing the `weth` view function and rely on the auto generated `WETH()` getter, unless a lowercase alias is required for a specific interface compatibility reason, in which case the rationale should be documented in the NatSpec.

Coinbase: Acknowledged.

Cantina Managed: Acknowledged.

3.1.4 Missing share slippage protection on Morpho vault deposits

Severity: Informational

Context: MorphoLendPolicy.sol#L170-L174

Description: The function `_onSingleExecutorExecute` from contract `MorphoLendPolicy` authorizes and executes deposits only in terms of `depositAssets`, without committing to or validating the amount of vault shares received. The execution path consumes the recurring allowance based on the asset amount and then builds a bare `deposit(assets, account)` call against the configured Morpho vault.

Since the vault deposit operation returns minted shares but the policy does not use that return value or perform any post-execution share-balance verification, the account can deposit the full authorized asset amount while receiving materially fewer shares if the vault share price moves adversely before execution. This leaves the recurring allowance consumed even though the effective deposit terms have deteriorated.

Recommendation: Consider extending the execution payload with a minimum acceptable shares bound and reject executions that would mint fewer shares than authorized. Consider enforcing that bound either through a pre-execution share preview or through a same-transaction post-execution share-balance delta check.

Coinbase: Acknowledged, will not address with onchain checks. Will discuss with offchain products which protect against using new vaults.

Cantina Managed: Acknowledged by the Coinbase team.

3.1.5 A top up may not bring user's ltv below `triggerLtv` threshold

Severity: Informational

Context: MorphoLoanProtectionPolicy.sol#L297-L303

Description: The `MorphoLoanProtectionPolicy` is a one shot policy which means that it can only be executed once for an account.

In `MorphoLoanProtectionPolicy._onPostExecute` function the `postTopUpLtv` is only being compared with liquidation ltv of Morpho market which means that any top up which doesn't push the account's ltv above `lltv` is valid and consumes the one shot top up opportunity of the account.

Ideally it should be enforced that the top up is performed with optimal top up amount which puts the account's ltv in a safer range. One way to approach this is to compare the `postTopUpLtv` with `LoanProtectionPolicyConfig.triggerLtv`.

```
  //...
  if (postTopUpLtv >= marketParams.lltv) revert();
+ if (postTopUpLtv >= config.triggerLtv) revert();
```

Coinbase: Acknowledged, will not address. We want to allow actions that prevent liquidation even if it doesn't make the loan particularly healthy.

Cantina Managed: Acknowledged by the Coinbase team.