CANTINA

Euler PR 111

Security Review

Cantina Managed review by:

Sujith Somraaj, Lead Security Researcher
Slowfi, Security Researcher

October 30, 2025

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**

3 Findings **4**
- 3.1 Low Risk . . . 4
  - 3.1.1 Default fee recipient can be set to zero address . . . 4
  - 3.1.2 Missing events for administrative state changes . . . 4
  - 3.1.3 Activation marks pool active before initialization completes . . . 5
- 3.2 Informational . . . 5
  - 3.2.1 Remove unused code . . . 5
  - 3.2.2 Improve code documentation . . . 5
  - 3.2.3 Internal function `delegateToManagementImpl` violates conventions . . . 6
  - 3.2.4 Do not use EVC sub-accounts as fee recipients . . . 6
  - 3.2.5 Admin address must not be an EVC sub-account . . . 6
  - 3.2.6 Protocol fee override can target non existent pool . . . 7

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

Euler Labs is a team of developers and quantitative analysts building DeFi applications for the future of finance.

From Oct 26th to Oct 28th the Cantina team conducted a review of euler-swap (PR 111) on commit hash 9287ae4f. The team identified a total of **9** issues:

Issues Found

Severity	Count	Fixed	Acknowledged
Critical Risk	0	0	0
High Risk	0	0	0
Medium Risk	0	0	0
Low Risk	3	3	0
Gas Optimizations	0	0	0
Informational	6	5	1
Total	**9**	**8**	**1**

3 Findings

3.1 Low Risk

3.1.1 Default fee recipient can be set to zero address

Severity: Low Risk

Context: EulerSwapProtocolFeeConfig.sol#L56

Description: The function `setDefaultRecipient` from contract `EulerSwapProtocolFeeConfig` assigns `defaultRecipient = recipient` without rejecting `address(0)`. If `defaultRecipient` is set to zero, any pool without an override may resolve to a zero recipient, which can burn fees or break downstream transfers depending on token behavior.

Also it is important to know that this breaks the flow from the previous implementation. Previously if protocol fee recipient is address(0), the fees are deposited into the euler account vs now they are burned. As a result, the account's NAV won't increase (as expected).

Recommendation: Consider to enforce a non-zero `recipient` in `setDefaultRecipient`. If you need a "reset" control path, consider to add an explicit `unsetDefaultRecipient()` (with a safe fallback recipient) or track a separate boolean sentinel rather than using `address(0)`.

Euler: Fixed in commit 9f5a2278.

Cantina Managed: Fix verified.

3.1.2 Missing events for administrative state changes

Severity: Low Risk

Context: EulerSwapProtocolFeeConfig.sol#L48-L65

Description: The functions `setAdmin`, `setDefault`, `setOverride`, and `removeOverride` from contract `EulerSwapProtocolFeeConfig` update critical configuration without emitting events (`src/EulerSwapProtocolFeeConfig.sol:48+`). The absence of events reduces on-chain observability for indexers, off-chain monitoring, audits, and incident response, and makes it harder to track configuration history per pool.

- The function `setAdmin` from contract `EulerSwapProtocolFeeConfig` changes the admin without an event.
- The function `setDefault` from contract `EulerSwapProtocolFeeConfig` updates the global default fee recipient/fee without an event.
- The function `setOverride` from contract `EulerSwapProtocolFeeConfig` sets a per-pool override without an event.
- The function `removeOverride` from contract `EulerSwapProtocolFeeConfig` clears a per-pool override without an event.

Recommendation: Consider to emit explicit events for each state transition, including previous and new values and indexed fields for efficient querying. For example:

```
event AdminUpdated(address indexed oldAdmin, address indexed newAdmin);
event DefaultUpdated(address indexed oldRecipient, address indexed newRecipient, uint64 oldFee, uint64 newFee);
event OverrideSet(address indexed pool, address indexed recipient, uint64 fee);
event OverrideRemoved(address indexed pool);
```

Emit the corresponding event in `setAdmin`, `setDefault`, `setOverride`, and `removeOverride`. Also consider to emit events during contract initialization to establish the baseline configuration in logs.

Euler: Fixed in commit 04215434.

Cantina Managed: Fix verified.

3.1.3 Activation marks pool active before initialization completes

Severity: Low Risk

Context: EulerSwapManagement.sol#L68

Description: The function `activate` from contract `EulerSwapManagement` sets `s.status = 1` at `src/EulerSwapManagement.sol:68` before completing all initialization. If status `1` represents "active," any external call made afterward (directly or via hooks) could observe the pool as active and enter flows that assume full initialization is finished.

Recommendation: Consider to mark the pool as "initializing" first (e.g., `s.status = 2`) and only set `s.status = 1` at the very end, after all external calls and state are finalized. Also consider to replace magic numbers with an `enum` for clarity and to enforce status checks (`require(s.status == Initializing/Inactive/Active)`) at critical entry points.

Euler: Fixed in commit 315466da.

Cantina Managed: Fix verified.

3.2 Informational

3.2.1 Remove unused code

Severity: Informational

Context: EulerSwapManagement.sol#L4-L7, EulerSwapManagement.sol#L22, EulerSwap.sol#L21-L24, FundsLib.sol#L10, UniswapHook.sol#L19, UniswapHook.sol#L21-L22

Description: Multiple files under the scope of the audit contain unused file imports, error declarations, or event declarations. This dead code will affect code quality. Consider deleting any unused file imports, error, and event declarations referenced in this finding.

Euler: Fixed in commit 6535ab75.

Cantina Managed: Fix verified.

3.2.2 Improve code documentation

Severity: Informational

Context: IEulerSwapCallee.sol#L5-L10, FundsLib.sol#L77

Description:

1. In `FundsLib.depositAssets()`, the inline documentation mentions that only E_ZeroShares is handled by setting the amount to zero, but the code actually manages both E_ZeroShares and ZeroShares errors.

```
- /// @dev If the deposit fails with E_ZeroShares error, it safely returns 0 (this happens with very
↪  small amounts).
+ /// @dev If the deposit fails with E_ZeroShares or ZeroShares error, it safely returns 0 (this happens
↪  with very small amounts).
```

2. All non-abstract contracts in the repository have inline documentation and a security contact before declaration, except the new `EulerSwapManagement` and `EulerSwap` contracts.

```
+ /// @title EulerSwapManagement contract
+ /// @custom:security-contact security@euler.xyz
+ /// @author Euler Labs (https://www.eulerlabs.com/)

      contract EulerSwapManagement is EulerSwapBase {
```

3. The function `eulerSwapCall()` from contract `IEulerSwapCallee` defines the flash-swap callback, but its parameters are sparsely documented. This can hinder integrators from implementing correct settlement logic, leading to subtle bugs. Consider to expand the NatSpec for eulerSwapCall to clarify integrators on received parameters.

Euler: Fixed in commit 8466db64.

Cantina Managed: Fix verified.

3.2.3 Internal function `delegateToManagementImpl` violates conventions

Severity: Informational

Context: EulerSwap.sol#L32-L39

Description: The internal function `delegateToManagementImpl()` violates standard Solidity conventions in two ways:

1. Naming Convention: Internal functions should be prefixed with an underscore _ to distinguish them from external/public functions clearly. The function is named `delegateToManagementImpl()` instead of `_delegateToManagementImpl()`.
2. File Organization: Internal and private functions are conventionally placed at the bottom of the contract, after all external and public functions.

Consider fixing the above-mentioned convention issue.

Euler: Fixed in commit f7528db3.

Cantina Managed: Fix verified.

3.2.4 Do not use EVC sub-accounts as fee recipients

Severity: Informational

Context: *(No context files were provided by the reviewer)*

Description: The functions `setDefault` and `setOverride` from contract `EulerSwapProtocolFeeConfig`, and the fee transfer paths in `SwapLib` (LP and protocol fee handling), do not remap EVC sub-accounts to their owners before ERC20 transfers. Unlike the registry's bond redemption flow, which calls `evc.getAccountOwner(recipient)` to resolve the owner, fee recipients are used as-is. As a result, configuring LP or protocol fee recipients to an EVC sub-account can direct fees to an address without control keys. Additionally, the activation guard only forbids `feeRecipient == eulerAccount` for the pool; other EVC sub-accounts are still permitted.

Recommendation: Consider to explicitly document that LP and protocol fee recipients must be externally owned addresses or controllable contract addresses, and should not be EVC sub-accounts. Include this constraint in deployment runbooks, admin UI validations, and configuration examples. Optionally, consider to note that the registry remaps bond recipients to owners, but fee flows currently do not, to avoid assumptions of parity.

Euler: Fixed in commit cfe12541.

Cantina Managed: Fix verified.

3.2.5 Admin address must not be an EVC sub-account

Severity: Informational

Context: EulerSwapProtocolFeeConfig.sol#L38-L50

Description: The function `setAdmin` from contract `EulerSwapProtocolFeeConfig` updates the `admin` without restricting it to a non-EVC sub-account. Subsequent admin-only calls use `onlyAdmin`, which invokes `_authenticateCallerWithStandardContextState(true)`. If the stored `admin` is an EVC sub-account, or if calls are routed through EVC with `onBehalfOfAccount` set to a sub-account, authentication can fail, effectively bricking administrative functions for this contract.

Recommendation: Consider to document that `admin` must be a controllable EOA or standard contract address, not an EVC sub-account. Include this in deployment runbooks and any admin tooling. Optionally, consider to validate in ops tooling that `newAdmin` is not an EVC sub-account (e.g., warn when `evc.getAccountOwner(newAdmin) != address(0)`).

Euler: Fixed in commit 0e0d9462.

Cantina Managed: Fix verified.

3.2.6 Protocol fee override can target non existent pool

Severity: Informational

Context: EulerSwapProtocolFeeConfig.sol#L61-L65

Description: The function `setOverride` from contract `EulerSwapProtocolFeeConfig` sets `overrides[pool]` without validating that `pool` is a real EulerSwap pool. This enables typos or stale addresses to create dead configuration entries that will never be used, complicating ops and audits.

Recommendation: Consider to verify the `pool` against the factory/registry before storing the override (e.g., `IEulerSwapFactory.isPool(pool)` or equivalent), and revert if the pool is unknown. Also consider to emit an event so indexers can track successful validations.

Euler: There may exist a use case for overriding the fees before the pool creation to have the exiting data at pool time creation.

Cantina Managed: Acknowledged by Euler team.