Security Assessment
Final Report

MEV Tax Hook

February 2025

Prepared for Balancer Labs

certora Balancer

Table of Contents

Project Summary **3**
Project Scope 3
Project Overview 3
Protocol Overview 3
Findings Summary 4
Severity Matrix 4
Informational Severity Issues **5**
I-01. Inconsistent priority gas price check across liquidity hooks and swap fee calculation 5
I-02. Unused FixedPoint library import 6
I-03. Inconsistent naming of isMevTaxExempt() function 6
Disclaimer **7**
About Certora **7**

certora Balancer

Project Summary

Project Scope

Project Name	Repository (link)	Latest Commit Hash	Platform
Balancer V3	balancer-v3-monorepo	767a6a1	EVM

Project Overview

This document describes the specification and verification of MEV Tax Hook using the Certora Prover and manual code review findings. The work was undertaken from 3 February 2025 to 7 February 2025.

The following contract list is included in our scope:

`pkg/pool-hooks/contracts/MevTaxHook.sol`

During the manual audit, the Certora team discovered issues in the Solidity contracts code, as listed on the following page.

Protocol Overview

The MEV Tax Hook enables Balancer pools to capture their own MEV through a dynamic swap fee based on transaction priority fees. It calculates tax by multiplying the priority fee by a configurable tax multiplier, which combines with the static base fee to capture MEV from high-priority transactions.

Findings Summary

The table below summarizes the findings of the review, including type and severity details.

Severity	Discovered	Confirmed	Fixed
Critical	-	-	-
High	-	-	-
Medium	-	-	-
Low	-	-	-
Informational	3		
Total	3		

Severity Matrix

Impact	High	Medium	High	Critical
	Medium	Low	Medium	High
	Low	Low	Low	Medium
		Low	Medium	High
		Likelihood		

Informational Severity Issues

I-01. Inconsistent priority gas price check across liquidity hooks and swap fee calculation

Description: The contract categorizes transactions as retail if `priorityGasPrice < threshold`, as seen in `_calculateSwapFeePercentage()`:

```
C/C++
if (priorityGasPrice < threshold || maxMevSwapFeePercentage < staticSwapFeePercentage) {
    return staticSwapFeePercentage;
}
```

However, `onBeforeAddLiquidity()` and `onBeforeRemoveLiquidity()` use an inclusive (`<=`) check:

```
C/C++
return kind == RemoveLiquidityKind.PROPORTIONAL || priorityGasPrice <=
_poolMevTaxThresholds[pool];
```

This inconsistency could cause unexpected behavior, where a transaction is treated as retail for liquidity actions but MEV for swaps.

Recommendation: Ensure consistency by updating liquidity hooks to use a strictly less than (`<`) check. This aligns the logic across the contract, preventing discrepancies.

Customer's response: At the threshold (==), the hook applies the static swap fee, making it effectively a retail trade. Using <= instead of < in the dynamic fee hook improves efficiency by returning earlier and skipping unnecessary calculations. Fixed in b0a7ad4.

I-02. Unused FixedPoint library import

Description: The `MevCaptureHook` contract imports the FixedPoint library:

```
Unset
import { FixedPoint } from "@balancer-labs/v3-solidity-utils/contracts/math/FixedPoint.sol";
```

However, the contract does not utilize any functions from the `FixedPoint` library, making the import as well as the `using ... for` directive redundant.

Recommendation: Remove the unused `FixedPoint` import and the `using FixedPoint for uint256;` statement to simplify the code.

Customer's response: Confirmed and fixed in 6c3c21c.

I-03. Inconsistent naming of isMevTaxExempt() function

Description: The function `isMevTaxExempt()` is inconsistently named compared to related functions `addMevTaxExemptSenders()` and `removeMevTaxExemptSenders()`. This discrepancy reduces readability and maintainability.

Recommendation: Rename `isMevTaxExempt()` to `isMevTaxExemptSender()`. This ensures naming consistency across the contract and improves code clarity.

Customer's response: Confirmed and fixed in b0a7ad4.

Disclaimer

Even though we hope this information is helpful, we provide no warranty of any kind, explicit or implied. The contents of this report should not be construed as a complete guarantee that the contract is secure in all dimensions. In no event shall Certora or any of its employees be liable for any claim, damages, or other liability, whether in an action of contract, tort, or otherwise, arising from, out of, or in connection with the results reported here.

About Certora

Certora is a Web3 security company that provides industry-leading formal verification tools and smart contract audits. Certora's flagship security product, Certora Prover, is a unique SaaS product that automatically locates even the most rare & hard-to-find bugs on your smart contracts or mathematically proves their absence. The Certora Prover plugs into your standard deployment pipeline. It is helpful for smart contract developers and security researchers during auditing and bug bounties.

Certora also provides services such as auditing, formal verification projects, and incident response.