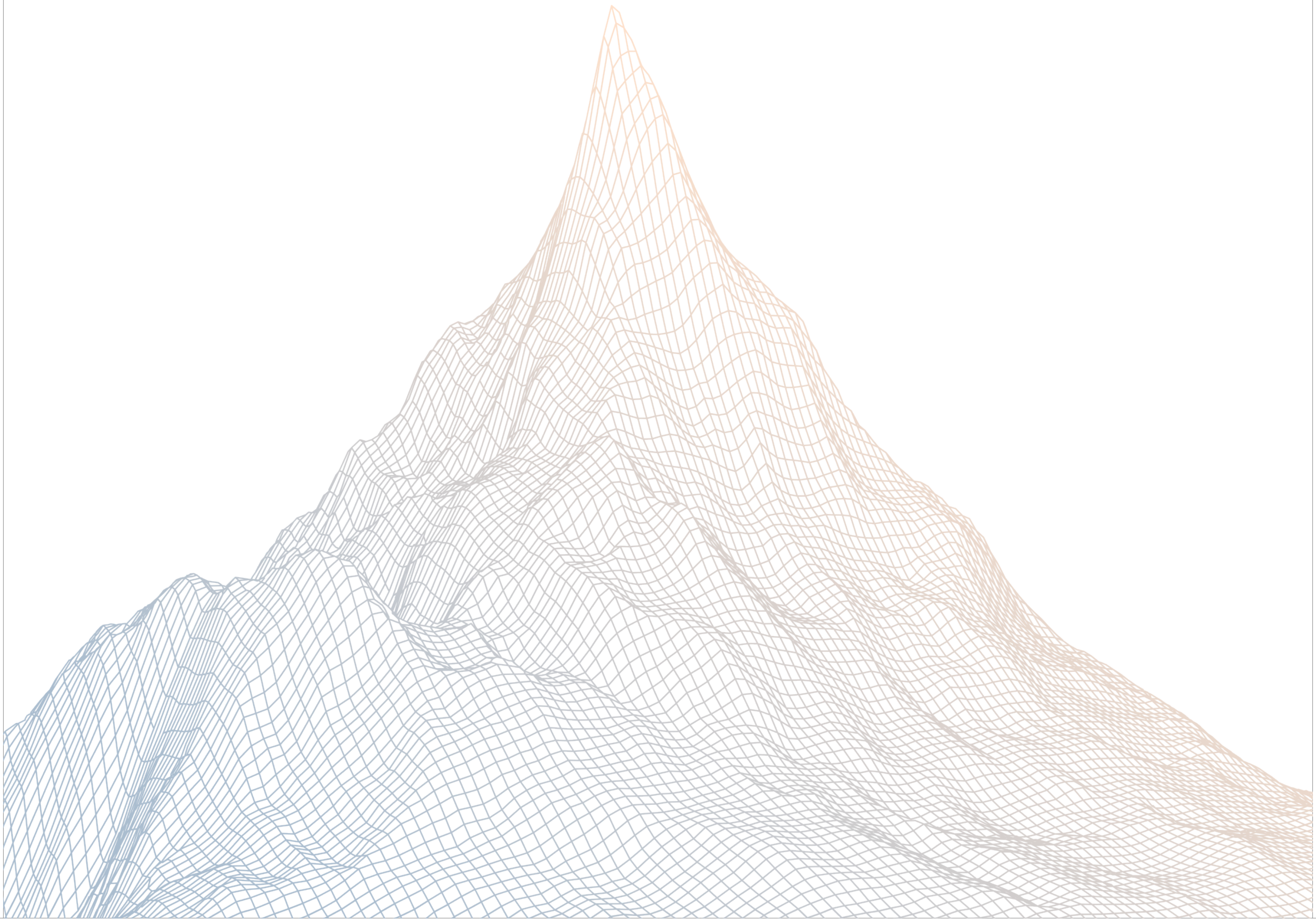
Zenith

Berachain

Smart Contract Security Assessment

VERSION 1.1



AUDIT DATES: July 10th to July 14th, 2025

AUDITED BY: ether_sky
zzykxx

Contents

1 Introduction

1.1 About Zenith

Zenith assembles auditors with proven track records: finding critical vulnerabilities in public audit competitions.

Our audits are carried out by a curated team of the industry's top-performing security researchers, selected for your specific codebase, security needs, and budget.

Learn more about us at https://zenith.security.

1.2 Disclaimer

This report reflects an analysis conducted within a defined scope and time frame, based on provided materials and documentation. It does not encompass all possible vulnerabilities and should not be considered exhaustive.

The review and accompanying report are presented on an "as-is" and "as-available" basis, without any express or implied warranties.

Furthermore, this report neither endorses any specific project or team nor assures the complete security of the project.

1.3 Risk Classification

SEVERITY LEVEL	IMPACT: HIGH	IMPACT: MEDIUM	IMPACT: LOW
Likelihood: High	Critical	High	Medium
Likelihood: Medium	High	Medium	Low
Likelihood: Low	Medium	Low	Low

2

Executive Summary

2.1 About Berachain

Berachain is a high-performance EVM-Identical Layer 1 blockchain utilizing Proof-of-Liquidity (PoL) and built on top of the modular EVM-focused consensus client framework BeaconKit.

2.2 Scope

The engagement involved a review of the following targets:

Target	contracts-internal
Repository	https://github.com/berachain/contracts-internal/
Commit Hash	`297a3795879b8103d6144541433d8ad604bce74e`
Files	`Changes in PR-23`

2.3 Audit Timeline

July 10th, 2025	Audit start
July 14th, 2025	Audit end
July 16th, 2025	Report published

2.4 Issues Found

SEVERITY	COUNT
Critical Risk	0
High Risk	0
Medium Risk	1
Low Risk	1
Informational	1
Total Issues	**3**

3

Findings Summary

ID	Description	Status
M-1	WBERAStakerVault can be subject to an inflation attack despite mitigations	Resolved
L-1	BGTIncentiveFeeCollector::claimFees() can be front-run with a deposit to the vault in order to capture profits	Acknowledged
I-1	WBERAStakerVault::completeWithdrawal() lacks whenNotPaused modifier	Resolved

4 Findings

4.1 Medium Risk

A total of 1 medium risk findings were identified.

[M-1] `WBERAStakerVault` can be subject to an inflation attack despite mitigations

`SEVERITY:` Medium	`IMPACT:` Medium
`STATUS:` Resolved	`LIKELIHOOD:` Low

Target

- BGTIncentiveFeeDeployer

Description:

The BGTIncentiveFeeDeployer contract deposits an initial amount of `WBERA` into the deployed vault in order to avoid inflation attacks.

It's still possible for an attacker to perform an inflation attack on the initial deposit itself:

1. Monitor the mempool for BGTIncentiveFeeDeployer deployment.
2. Calculate the address at which WBERAStakerVault will be deployed and transfer `10e18 WBERA` to said address before the contract is deployed.
3. The constructor of BGTIncentiveFeeDeployer is executed, which makes the first deposit in the pool. Because `10e18 WBERA` already exists in the contract the vault will mint `0` shares.

The initial deposit of `10e18` is lost and the attack can turn profitable if more users deposit.

Recommendations:

In the constructor of BGTIncentiveFeeDeployer require the total of shares to be equal to `BGTIncentiveFeeDeployer` after the deposit. By doing this the deployment will fail if an attacker attempts this attack, losing the funds:

```
require(wberaStakerVault.totalSupply() == INITIAL_DEPOSIT_AMOUNT);
```

Berachain: Resolved with @64d3d9...

Zenith: Verified.

4.2 Low Risk

A total of 1 low risk findings were identified.

[L-1] `BGTIncentiveFeeCollector::claimFees()` can be front-run with a deposit to the vault in order to capture profits

SEVERITY: Low	IMPACT: Low
STATUS: Acknowledged	LIKELIHOOD: Medium

Target

- BGTIncentiveFeeCollector

Description:

The BGTIncentiveFeeCollector::claimFees() transfers `WBERA` from the caller to the vault. This results in the value of each share increasing instantly.

Users knowing this can frontrun a call to BGTIncentiveFeeCollector::claimFees() by depositing `WBERA` in order to get shares whose value will instantly increase, then they can schedule a withdrawal.

Recommendations:

This is already mitigated because instant withdrawals are not possible but will still result in the possibility of capturing value that should belong to honest users.

Making sure `payoutAmount` is a low value helps further mitigating the issue, this is non-trivial to fix without heavy changes.

Berachain: Acknowledged.

4.3 Informational

A total of 1 informational findings were identified.

[I-1] `WBERAStakerVault::completeWithdrawal()` lacks `whenNotPaused` modifier

`SEVERITY`: Informational	`IMPACT`: Informational
`STATUS`: Resolved	`LIKELIHOOD`: Low

Target

- WBERAStakerVault

Description:

The function WBERAStakerVault:: completeWithdrawal() can be executed even when the contract is paused.

Recommendations:

Add a `whenNotPaused` modifier to WBERAStakerVault:: completeWithdrawal().

Berachain: Resolved with @64d3d90a....

Zenith: Verified.