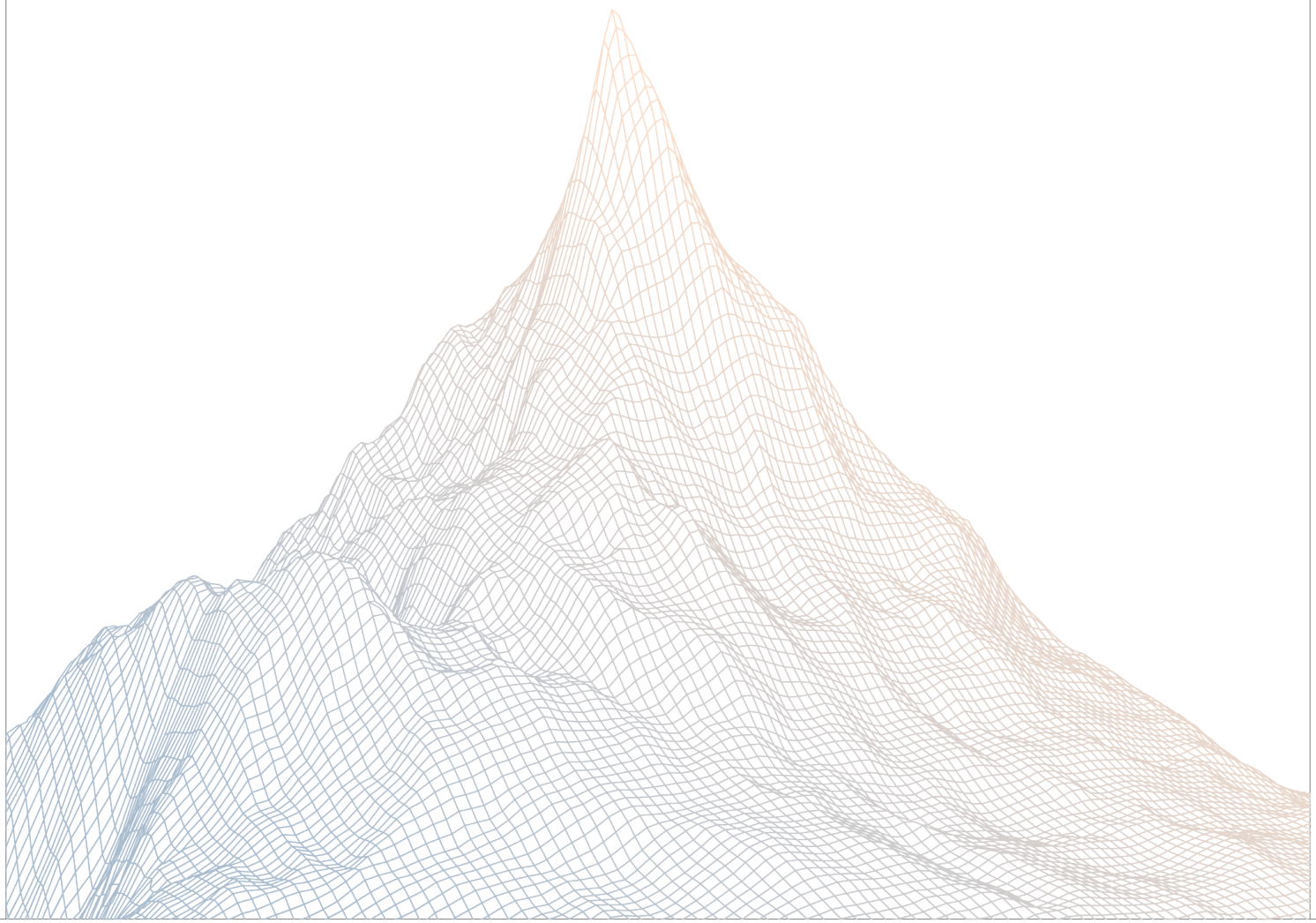
Zenith

Berachain

Smart Contract Security Assessment

VERSION 1.1



AUDIT DATES: December 16th to December 19th, 2025

AUDITED BY: Matte
ether_sky

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/pull/71
Commit Hash	`d8c01b934f6de3a3a509ec1169255f7ec066e80a`
Files	`Changes in PR-71`

2.3 Audit Timeline

December 16, 2025	Audit start
December 19, 2025	Audit end
January 8, 2026	Report published

2.4 Issues Found

SEVERITY	COUNT
Critical Risk	0
High Risk	0
Medium Risk	1
Low Risk	0
Informational	1
Total Issues	**2**

3

Findings Summary

ID	Description	Status
M-1	Claiming fees may revert in certain cases	Resolved
I-1	There is an unnecessary approval in InfraredBeraAdapter	Resolved

4

Findings

4.1 Medium Risk

A total of 1 medium risk findings were identified.

[M-1] Claiming fees may revert in certain cases

SEVERITY: Medium	IMPACT: Medium
STATUS: Resolved	LIKELIHOOD: Medium

Target

- InfraredBeraAdapter.sol

Description:

In `BGTIncentiveFeeCollector`, `WBERA` is distributed to `vaults` based on their `staked amounts`.

- BGTIncentiveFeeCollector.sol#L254

```
function _splitAmount(uint256 amount) internal view returns (uint256[]
    memory amounts) {
    uint256 distributed;
    for (uint256 i; i < len + 1;) {
        uint256 part = (amount * stakes[i]) / totalStake;
        amounts[i] = part;
        distributed += part;
        unchecked {
            ++i;
        }
    }

    return amounts;
}
```

For `vaults` with very small `stakes`, the distributed `amount` can be as low as `1-2 wei`.

These amounts are then passed to the `stake` function of `InfraredBeraAdapter`, which calls the `mint` function on `InfraredBERA`.

- InfraredBeraAdapter.sol#L33

```
function stake(uint256 amount) external returns (uint256) {
    wbera.transferFrom(msg.sender, address(this), amount);
    wbera.approve(address(wbera), amount);
    wbera.withdraw(amount);

    uint256 shares = IInfraredBera(INFRARED_BERA_ADDR).mint{ value: amount
    }(msg.sender);

    emit Stake(amount, shares);
    return shares;
}
```

Since the `share price` in `InfraredBERA` increases over time, `minting` with such small `deposit amounts` can result in zero `shares` being `minted`.

When this happens, the transaction reverts, preventing fees from being claimed.

```
function mint(address receiver) public payable returns (uint256 shares) {
    compound();

    uint256 d = deposits;
    uint256 ts = totalSupply();

    uint256 amount = msg.value;
    _deposit(amount);

    shares = (d != 0 && ts != 0) ? (ts * amount) / d : amount;
@->    if (shares == 0) revert Errors.InvalidShares();
    _mint(receiver, shares);

    emit Mint(receiver, amount, shares);
}
```

Recommendations:

Check the `minted share` amount using `previewMint` function and skip the `mint` call if the amount is zero.

Berachain: Resolved with `@91c34cc15c ...`

Zenith: Verified.

4.2 Informational

A total of 1 informational findings were identified.

[I-1] There is an unnecessary approval in InfraredBeraAdapter

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

Target

- InfraredBeraAdapter.sol

Description:

Below approval is unnecessary.

- InfraredBeraAdapter.sol#L30

```
function stake(uint256 amount) external returns (uint256) {
    wbera.transferFrom(msg.sender, address(this), amount);
▷   wbera.approve(address(wbera), amount);
    wbera.withdraw(amount);

    uint256 shares = IInfraredBera(INFRARED_BERA_ADDR).mint{ value:  amount
    }(msg.sender);

    emit Stake(amount, shares);
    return shares;
}
```

Recommendations:

Remove that line in the `stake` function.

Berachain: Resolved with `@26a350ee67 ...`

Zenith: Verified.