



# LI.FI

---

## LI.FI Security Review

---

SwapperV2.sol(v1.1.0)

**Security Researcher**

Sujith Somraaj ([somraajsujith@gmail.com](mailto:somraajsujith@gmail.com))

**Report prepared by:** Sujith Somraaj

June 29, 2025

Contents

1 About Researcher 2

2 Disclaimer 2

3 Scope 2

4 Risk classification 2

4.1 Impact . . . . . 2

4.2 Likelihood . . . . . 3

4.3 Action required for severity levels . . . . . 3

5 Executive Summary 3

6 Findings 4

6.1 Informational . . . . . 4

6.1.1 Remove unchecked loop increments . . . . . 4

# 1 About Researcher

Sujith Somraaj is a distinguished security researcher and protocol engineer with over eight years of comprehensive experience in the Web3 ecosystem.

In addition to working as a Security researcher at Spearbit, Sujith is also the security researcher and advisor for leading bridge protocol LI.FI and also is a former founding engineer and current CISO at Superform, a yield aggregator with over \$170M in TVL.

Sujith has experience working with protocols / funds including Edge Capital, Berachain, Optimism, Sonic, Monad, Blast, ZkSync, Decent, Drips, SuperSushi Samurai, DistrictOne, Omni-X, Centrifuge, Superform-V2, Tea.xyz, Paintswap, Bitcorn, Sweep n' Flip, Byzantine Finance, Variational Finance, Satsbridge, Earthfast and Angles

Learn more about Sujith on [sujithsomraaj.xyz](https://sujithsomraaj.xyz) or on [cantina.xyz](https://cantina.xyz)

## 2 Disclaimer

Note that this security audit is not designed to replace functional tests required before any software release, and does not give any warranties on finding all possible security issues of that given smart contract(s) or blockchain software. i.e., the evaluation result does not guarantee against a hack (or) the non existence of any further findings of security issues. As one audit-based assessment cannot be considered comprehensive, I always recommend proceeding with several audits and a public bug bounty program to ensure the security of smart contract(s). Lastly, the security audit is not an investment advice.

This review is done independently by the reviewer and is not entitled to any of the security agencies the researcher worked / may work with.

## 3 Scope

- src/Helpers/SwapperV2.sol(v1.1.0)

## 4 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

### 4.1 Impact

- High** leads to a loss of a significant portion (>10%) of assets in the protocol, or significant harm to a majority of users.
- Medium** global losses <10% or losses to only a subset of users, but still unacceptable.
- Low** losses will be annoying but bearable — applies to things like grieving attacks that can be easily repaired or even gas inefficiencies.

## 4.2 Likelihood

- High** almost certain to happen, easy to perform, or not easy but highly incentivized
- Medium** only conditionally possible or incentivized, but still relatively likely
- Low** requires stars to align, or little-to-no incentive

## 4.3 Action required for severity levels

- Critical** Must fix as soon as possible (if already deployed)
- High** Must fix (before deployment if not already deployed)
- Medium** Should fix
- Low** Could fix

## 5 Executive Summary

Over the course of 2 hours in total, [LI.FI](#) engaged with the [researcher](#) to audit the contracts described in section 3 of this document ("scope").

In this period of time a total of 1 issues were found. This review focussed only on the changes made from the previous version, not the code on its entirety.

Project Summary	
Project Name	LI.FI
Repository	<a href="#">lifinance/contracts</a>
Commit	<a href="#">4af7030d</a>
Audit Timeline	June 29, 2025
Methods	Manual Review
Documentation	Medium-High
Test Coverage	Medium-High

Issues Found	
Critical Risk	0
High Risk	0
Medium Risk	0
Low Risk	0
Gas Optimizations	0
Informational	1
<b>Total Issues</b>	<b>1</b>

## 6 Findings

### 6.1 Informational

#### 6.1.1 Remove unchecked loop increments

**Context:** [SwapperV2.sol#L338](#)

**Description:** Unchecked loop increment does not offer any gas benefits like previous compiler versions. Hence the use of unchecked loop increments can be removed to improve code quality.

**Recommendation:** Consider updating the function as follows:

```
function _refundLeftovers(
    LibSwap.SwapData[] calldata _swaps,
    address payable _leftoverReceiver,
    uint256[] memory _initialBalances,
    uint256 _nativeReserve
) private {
    ....
-   for (uint256 i = 0; i < numSwaps; ) {
+   for (uint256 i; i < numSwaps; ++i)
        ....
-       unchecked {
-           ++i;
-       }
    }
    ....
}
```

**LI.FI:** Fixed in [89548cdd8f3ea7eb6ab26aae2ca3611808d83e80](#)

**Researcher:** Verified fix