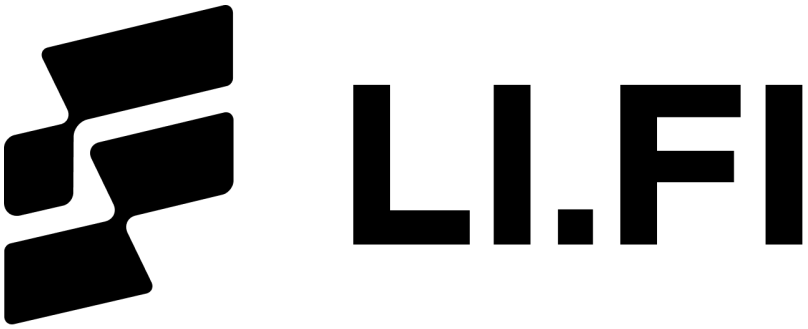

LI.FI Security Review

NEARIntentsFacet.sol(v1.0.0)

Security Researcher

Sujith Somraaj (somraajsujith@gmail.com)

Report prepared by: Sujith Somraaj

December 16, 2025

Contents

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 Lead 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 security advisor at Superform, a yield aggregator with over $170M in TVL.

Sujith has experience working with protocols / funds including Coinbase, Uniswap, Layerzero, Edge Capital, Berachain, Optimism, Ondo, 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, Rova, Horizen, Earthfast and Angles

Learn more about Sujith on sujithsomraaj.xyz or on 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/Facets/NEARIntentsFacet.sol(1.0.0)

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 griefing 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 6 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 5 issues were found.

Project Summary	
Project Name	LI.FI
Repository	lifinance/contracts
Commit	4696d22
Audit Timeline	December 15, 2025
Methods	Manual Review
Documentation	High
Test Coverage	Medium-High

Issues Found	
Critical Risk	0
High Risk	0
Medium Risk	0
Low Risk	0
Gas Optimizations	2
Informational	3
Total Issues	**5**

6 Findings

6.1 Gas Optimization

6.1.1 Inline event emission to eliminate unnecessary internal function call

Context: NEARIntentsFacet.sol#L241

Description: The `_startBridge()` function delegates event emission to a separate internal function `_emitEvents()`.

This introduces unnecessary function call overhead including JUMP operations, stack frame setup, and parameter passing. The `_emitEvents()` function is only called once and serves no abstraction purpose.

Recommendation: Inline the event emission directly into `_startBridge()` function to save ~40 GAS per call.

LI.FI: Fixed in 4e5b2ec

Researcher: Verified fix.

6.1.2 Remove unnecessary stack variables

Context: NEARIntentsFacet.sol#L282, NEARIntentsFacet.sol#L193

Description:

1. The `_verifySignature()` function creates a temporary boolean variable, isNonEVM, that is only used once in a ternary expression. This adds an unnecessary stack variable allocation and a DUP operation, consuming approximately 2-3 gas per invocation.
2. The `swapAndStartBridgeTokensViaNEARIntents()` funtion creates a temporary uint256 variable, positiveSlippage that is only used once. This could also be removed.

Recommendation: Consider removing the temporary variable:

```
- uint256 positiveSlippage = actualAmountAfterSwap - _bridgeData.minAmount;
   LibAsset.transferAsset(
     _bridgeData.sendingAssetId,
     payable(_nearData.refundRecipient),
-    positiveSlippage
+    actualAmountAfterSwap - _bridgeData.minAmount
     );

- bool isNonEVM = _bridgeData.receiver == NON_EVM_ADDRESS;
- bytes32 receiverHash = isNonEVM
+ bytes32 receiverHash = _bridgeData.receiver == NON_EVM_ADDRESS
```

LI.FI: Fixed in 4e5b2ec. The isNonEVM variable was successfully removed, but the positiveSlippage variable was kept as removing it causes a "Stack too deep" compilation error. The variable is necessary for managing stack depth in the function.

Researcher: Verified fix.

6.2 Informational

6.2.1 Misleading variable name `receiverHash`

Context: NEARIntentsFacet.sol#L284

Description: The variable receiverHash in the `_verifySignature()` function is misleadingly named. The variable does not contain a hash but rather a bytes32-encoded representation of the receiver address.

Recommendation: Rename the variable to accurately reflect its purpose:

```
- bytes32 receiverHash = isNonEVM
+ bytes32 receiverBytes32 = isNonEVM
```

LI.FI: Fixed in 4e5b2ec

Researcher: Verified fix.

6.2.2 Fee-on-Transfer tokens permanently consume quote IDs while underfunding bridge

wr**Context:** NEARIntentsFacet.sol#L231

Description: When users try to bridge fee-on-transfer tokens (such as SafeMoon, PAXG, etc.) via the NEAR Intents facet, the transaction fails, but the quote ID is permanently used up. This happens because the contract marks the quote as used without validating the destination balance.

Recommendation: Add an explicit warning in the contract documentation about this behavior to avoid bridging fee-on-transfer tokens using this facet.

LI.FI: Fixed in 4e5b2ec

Researcher: Verified fix.

6.2.3 Redundant `minAmountOut` parameter in `NearIntentsData`

Context: NEARIntentsFacet.sol#L53

Description: The **minAmountOut** field in `NearIntentsData` struct is included in the signature but never used or validated in the contract.

Recommendation: Consider removing it from the struct if unused, or if required for off-chain tracking, emit it in the **NEARIntentsBridgeStarted** event.

LI.FI: Fixed in 4e5b2ec

Researcher: Verified fix.