LI.FI Security Review

PolymerCCTP(v2.0.0)

Security Researcher

Sujith Somraaj (somraajsujith@gmail.com)

Report prepared by: Sujith Somraaj

February 16, 2026

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, Aztec, 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, Buck Labs 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/PolymerCCTPFacet.sol(v2.0.0)

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 2 hours in total, LI.FI engaged with the researcher to audit the contracts described in section 3 of this document ("scope"). This is a differential review focussed on the changes from previous version.

Changes Reviewed:

- Adding Monad chain ID
- PolymerCCTPFacet Uses Wrong ReentrancyGuard Enabling Cross-Facet Reentry
- When bridging to solana through CCTP the depositForBurn() call must use the mintRecipient as the recipient ATA, not the raw solana wallet address.

In this period of time a total of 1 issues were found.

Project Summary	
Project Name	LI.FI
Repository	lifinance/contracts
Commit	7f8a0ef
Audit Timeline	February 16, 2026
Methods	Manual Review
Documentation	High
Test Coverage	High

Issues Found	
Critical Risk	0
High Risk	0
Medium Risk	0
Low Risk	0
Gas Optimizations	0
Informational	1
Total Issues	**1**

6 Findings

6.1 Informational

6.1.1 Missing zero-value validation for `solanaReceiverATA` in non-evm bridge path

Context: PolymerCCTPFacet.sol#L218

Description: In `_startBridge()`, when bridging to a non-EVM chain (`_bridgeData.receiver == NON_EVM_ADDRESS`), the code validates that `_polymerData.nonEVMReceiver != bytes32(0)` but does not validate `_polymerData.solanaReceiverATA`. When the destination chain is Solana, mintRecipient is set to `_polymerData.solanaReceiverATA`, which could be `bytes32(0)`.

However, a comment states that TokenMessenger enforces mintRecipient != bytes32(0), so no explicit check is needed for solanaReceiverATA. However, this creates an inconsistency: nonEVMReceiver is validated at the facet level (reverting with InvalidReceiver) while solanaReceiverATA relies on the downstream TokenMessenger revert.

Both would ultimately fail in `depositForBurn()`, but with different error messages and at different call-stack depths.

Recommendation: Add an explicit zero-value check for solanaReceiverATA when the destination chain is Solana, reverting with InvalidConfig to remain consistent with the pattern established in EcoFacet.

Keep the existing nonEVMReceiver check with InvalidReceiver as-is.

```
bool isSolanaDestination = destinationChainId == LIFI_CHAIN_ID_SOLANA;
bytes32 mintRecipient = isSolanaDestination
      ? _polymerData.solanaReceiverATA
      : _polymerData.nonEVMReceiver;

if (mintRecipient == bytes32(0)) {
  if (isSolanaDestination) revert InvalidConfig();
  revert InvalidReceiver();
}
```

LI.FI: Fixed in 59d992d

Researcher: Verified fix.