PALADIN
BLOCKCHAIN SECURITY

Smart Contract Security Assessment

Preliminary Report

For LayerZero (VaultComposerSync)

06 August 2025

paladinsec.co

info@paladinsec.co

Table of Contents

Disclaimer

Paladin Blockchain Security ("Paladin") has conducted an independent audit to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the codes that were provided for the scope of this audit. This audit report does not constitute agreement, acceptance or advocation for the Project that was audited, and users relying on this audit report should not consider this as having any merit for financial advice in any shape, form or nature. The contracts audited do not account for any economic developments that may be pursued by the Project in question, and that the veracity of the findings thus presented in this report relate solely to the proficiency, competence, aptitude and discretion of our independent auditors, who make no guarantees nor assurance that the contracts are completely free of exploits, bugs, vulnerabilities or deprecation of technologies. Further, this audit report shall not be disclosed nor transmitted to any persons or parties on any objective, goal or justification without due written assent, acquiescence or approval by Paladin.

All information provided in this report does not constitute financial or investment advice, nor should it be used to signal that any persons reading this report should invest their funds without sufficient individual due diligence regardless of the findings presented in this report. Information is provided 'as is', and Paladin is under no covenant to the completeness, accuracy or solidity of the contracts audited. In no event will Paladin or its partners, employees, agents or parties related to the provision of this audit report be liable to any parties for, or lack thereof, decisions and/or actions with regards to the information provided in this audit report.

Cryptocurrencies and any technologies by extension directly or indirectly related to cryptocurrencies are highly volatile and speculative by nature. All reasonable due diligence and safeguards may yet be insufficient, and users should exercise considerable caution when participating in any shape or form in this nascent industry.

The audit report has made all reasonable attempts to provide clear and articulate recommendations to the Project team with respect to the rectification, amendment and/or revision of any highlighted issues, vulnerabilities or exploits within the contracts provided. It is the sole responsibility of the Project team to sufficiently test and perform checks, ensuring that the contracts are functioning as intended, specifically that the functions therein contained within said contracts have the desired intended effects, functionalities and outcomes of the Project team.

Paladin retains the right to re-use any and all knowledge and expertise gained during the audit process, including, but not limited to, vulnerabilities, bugs, or new attack vectors. Paladin is therefore allowed and expected to use this knowledge in subsequent audits and to inform any third party, who may or may not be our past or current clients, whose projects have similar vulnerabilities. Paladin is furthermore allowed to claim bug bounties from third-parties while doing so.

1 Overview

This report has been prepared for LayerZero's VaultComposerSync contracts on the Ethereum network. Paladin provides a user-centred examination of the smart contracts to look for vulnerabilities, logic errors or other issues from both an internal and external perspective.

1.1 Summary

Project Name	LayerZero
URL	https://layerzero.network/
Platform	Ethereum
Language	Solidity
Preliminary Contracts	https://github.com/LayerZero-Labs/devtools/commit/4ccff1f36cadb29562e512b8750a96416fb546c5
Resolution #1	

1.2 Contracts Assessed

Name	Contract	Live Code Match
`VaultComposerSync`		

1.3 Findings Summary

Severity	Found	Resolved	Partially Resolved	Acknowledged (no change made)
High	0	-	-	-
Medium	0	-	-	-
Low	0	-	-	-
Informational	4	4	-	-
Total	**4**	**4**	-	-

Classification of Issues

Severity	Description
High	Exploits, vulnerabilities or errors that will certainly or probabilistically lead towards loss of funds, control, or impairment of the contract and its functions. Issues under this classification are recommended to be fixed with utmost urgency.
Medium	Bugs or issues with that may be subject to exploit, though their impact is somewhat limited. Issues under this classification are recommended to be fixed as soon as possible.
Low	Effects are minimal in isolation and do not pose a significant danger to the project or its users. Issues under this classification are recommended to be fixed nonetheless.
Informational	Consistency, syntax or style best practices. Generally pose a negligible level of risk, if any.

1.3.1 VaultComposerSync

ID	Severity	Summary	Status
01	INFO	Refund will revert if `msg.value > 0` in `_send`	RESOLVED
02	INFO	Unnecessary share token approval	RESOLVED
03	INFO	`quoteSend` does not take global limits into consideration	RESOLVED
04	INFO	Typographical issues	RESOLVED

2 Findings

2.1 VaultComposerSync

`VaultComposerSync` is a smart contract that enables seamless, cross-chain deposits and redemptions for ERC4626-compliant vaults using LayerZero's Omnichain Fungible Token (OFT) protocol. It manages asset and share transfers between chains, incorporates slippage protection, and provides automatic refund mechanisms for failed operations, allowing users to interact with vaults synchronously and securely across multiple blockchains using LayerZero.

2.1.1 Privileged Functions

- `lzCompose [ENDPOINT]`
- `handleCompose [Contract itself]`

2.1.2 Issues & Recommendations

Issue #01	Refund will revert if `msg.value > 0` in `_send`
Severity	INFORMATIONAL
Description	Within `_send`, there is a check to see if `msg.value > 0` (L307). This check reverts with the same error as the check for a minimum `msg.value` in `handleCompose`.
Recommendation	Consider using another error inside `_send` to not clash with the one in `handleCompose`.
Resolution	RESOLVED

Issue #02	Unnecessary share token approval
Severity	INFORMATIONAL

Description	The constructor contains an unnecessary approval of share tokens to the vault, which is redundant since the share token is the vault itself.
	
	The contract enforces that `SHARE_ERC20 == address(VAULT)` on lines 67-69:
	`if (SHARE_ERC20 != address(VAULT)) {`
	`  revert ShareTokenNotVault(SHARE_ERC20, address(VAULT));`
	`}`
	
	so the following approval to `_vault` is unnecessary:
	`/// @dev Approve the vault to spend the share and asset tokens held by this contract`
	`IERC20(SHARE_ERC20).approve(_vault, type(uint256).max);`
	`IERC20(ASSET_ERC20).approve(_vault, type(uint256).max);`

Recommendation	Remove the unnecessary share token approval.
Resolution	RESOLVED

Issue #03	quoteSend does not take global limits into consideration
Severity	INFORMATIONAL
Description	`quoteSend()` calculates LayerZero messaging fees using `previewDeposit()` and `previewRedeem()` only. It does not verify that the requested `_vaultInAmount` is within the limits returned by `VAULT.maxDeposit(address(this))` and `VAULT.maxRedeem(address(this))`. If the composer's own capacity is already exhausted (per-address cap, global TVL cap, pause, epoch throttle, etc.), the function will still quote a non-zero fee even though the subsequent `deposit()` / `redeem()` call inside `lzCompose` will revert.
Recommendation	If this is desired behavior, consider commenting on the fact that this function does not take into consideration the maximum limit of the underlying ERC4626. If the team wishes to take the maximum limits into consideration, consider checking the `_vaultInAmount` against the `maxDeposit`/ `maxRedeem` as well.
Resolution	RESOLVED

Issue #04	Typographical issues
Severity	INFORMATIONAL

Description	The `_redeemAndSend` and `_depositAndSend` do not clearly state that the `mintAmountLD` must be in shares or assets depending on the path. Consider adding an explicit comment like:
	`NOTE: _sendParam.minAmountLD must be denominated in SHARES for slippage protection`
	
	and
	`NOTE: _sendParam.minAmountLD must be denominated in ASSETS for slippage protection`

Recommendation	Consider fixing the issues.
Resolution	RESOLVED

PALADIN
BLOCKCHAIN SECURITY