PALADIN
BLOCKCHAIN SECURITY

Smart Contract Security Assessment

Final Report

For LayerZero (OVault Yield.xyz Integration)

19 November 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 OVault 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/monorepo-internal/pull/1442/commits/f2dfe23d0ea5556b9abd77bd6eda5ce45f888bbb
Resolution #1	https://github.com/LayerZero-Labs/monorepo-internal/commit/5cae938dc156aa7a8b45b5291873e4139660c664

1.2 Contracts Assessed

- `YieldXyzOVaultComposerSync`
- `YieldXyzOVaultComposerSyncNative`

1.3 Findings Summary

Severity	Found	Resolved	Partially Resolved	Acknowledged (no change made)
High	0	-	-	-
Medium	0	-	-	-
Low	1	1	-	-
Informational	1	-	-	1
Total	**2**	**1**	-	1

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 YieldXyzOVaultComposerSync

ID	Severity	Summary	Status
01	LOW	Unhandled edge case causes funds to get stuck in composer or `lzCompose` to revert	✓ RESOLVED

1.3.2 YieldXyzOVaultComposerSyncNative

ID	Severity	Summary	Status
02	INFO	Missing `AssetOFTTokenNotNative` error arguments	ACKNOWLEDGED

2 Findings

2.1 YieldXyzOVaultComposerSync

`YieldXyzOVaultComposerSync` adapts LayerZero's `VaultComposerSync` to Yield.xyz allocator vaults by ensuring the asset OFT's token matches the vault's `underlying()` token. It sets the correct underlying token as the composer's asset token, wires in the asset/share OFTs, and pre-approves the vault and OFT adapter to pull underlying when needed.

All deposit, redeem, and cross-chain flows then use this underlying asset as the canonical ERC20 for the vault.

2.1.1 Privileged Functions

- `lzCompose[endpoint]`
- `handleCompose[address(this)]`

2.1.2 Issues & Recommendations

Issue #01	**Unhandled edge case causes funds to get stuck in composer or `lzCompose` to revert**
Severity	LOW SEVERITY

Description	If `config.hasCoolDown` is set to true, then the OVault will receive the strategy token and not the redeemed underlying token, so the slippage check will always fail if `_minAmountLD` is set to non-0.
	
	`if (config.hasCooldown) {`
	`  IERC20(address(strategy)).safeTransfer(receiver, assets);`
	`  } else {`
	`  strategy.redeem(assets, receiver, address(this));`
	`}`
	
	If `_minAmountLD` is set to 0, `assetAmountReceived` will be 0 if the strategy token is received, then the slippage check will pass and the transaction will not revert because but the strategy token will be stuck in the composer and the user will receive 0 `amountLd`.
	
	`uint256 assetAmountReceived = postAssetBalance - preAssetBalance;`
	`_assertSlippage(assetAmountReceived, _sendParam.minAmountLD);`
	`[..]`
	`function _assertSlippage(uint256 _amountLD, uint256 _minAmountLD) internal view virtual {`
	`  if (_amountLD < _minAmountLD) revert SlippageExceeded(_amountLD, _minAmountLD);`
	`}`

Recommendation	One way to fix this is to revert if `assetAmountReceived` is 0. This way, even if `minAmountLD` is 0, it will not pass the slippage check and revert.
Resolution	RESOLVED

2.2 YieldXyzOVaultComposerSyncNative

`YieldXyzOVaultComposerSyncNative` is the native-asset version, used when the asset OFT represents ETH and the vault's `underlying()` is a wrapped-ETH ERC20. It verifies that the asset OFT is truly native, sets the vault's underlying as the ERC20 asset, and pre-approves the vault to pull it.

All native inflows are wrapped into this ERC20 for deposits, and all outbound sends unwrap it back to ETH before bridging.

2.2.1 Privileged Functions

- `lzCompose[endpoint]`
- `handleCompose[address(this)]`

2.2.2 Issues & Recommendations

Issue #02	Missing `AssetOFTTokenNotNative` error arguments
Severity	INFORMATIONAL
Description	When `_initializeAssetToken()` in `YieldXyzOVaultComposerSyncNative` detects that `ASSET_OFT` is not a native OFT (`token() != address(0)`), it reverts with `AssetOFTTokenNotNative()` with no parameters. This makes it harder to debug misconfigurations, since it cannot be determined which OFT or token caused the failure, unlike `AssetTokenNotVaultAsset` in the non-native composer which already includes both conflicting addresses.
Recommendation	Extend `AssetOFTTokenNotNative` to include at least the `ASSET_OFT` address and the `token()` value (e.g. `error AssetOFTTokenNotNative(address assetOFT, address token)`), and pass those when reverting. This keeps the behavior the same while making configuration mistakes much easier to diagnose.
Resolution	ACKNOWLEDGED


PALADIN
BLOCKCHAIN SECURITY