YieldNest Max Vault Withdrawer and Lib Logic Audit Report

Feb 2025

Prepared for yieldnest.finance by NFR Audits
nofrontrunning@gmail.com

NFR AUDITS

Audit Summary

Scope

https://github.com/yieldnest/yieldnest-vault/tree/eth-max-vault commit 427e63649531455856bda863f4a1ed3070af1b0d

- src/BaseVault.sol
- src/library/VaultLib.sol
- src/library/AsyncWithdrawalLib.sol (used by Withdrawer)
- src/library/OriginWithdrawalLib.sol (library handling OETH withdrawals)
- src/module/Provider.sol (updated)
- src/withdraws/Withdrawer.sol (handles ETH derivative withdrawals)
- src/withdraws/BaseWithdrawer.sol

https://github.com/yieldnest/yieldnest-vault/blob/bnb-max-vault/src/withdraws/Withdrawer.sol commit 41b1deb585623141bf1d084b729bdd058ce2ab66

- src/withdraws/Withdrawer.sol

Codebase Overview

This audit contains mostly two parts, refactored vault logic into the VaultLib library and the withdrawer contract and libraries to handle withdraw functions. The VaultLib abstracts some shared logic and functions from the vault system. The Withdrawer works with the AsyncWithdrawalLib and OriginWithdrawalLib to handle asset accounting for queued async withdraws as well as withdraws from the Origin protocol.

The codebase is well-written, documented, and organized. The code structure is solid, and the quality is good. We recommend fixing the issues identified in this report before deployment. It will also be beneficial to set up off-chain monitoring of the system after deployment, especially for critical events and state updates. Further audits are highly recommended if new updates are made to the codebase in the future.

Auditing Methods

The codebase is manually audited line by line. Auto detectors and static analysers are also used to ensure best coverage of ulnerabilities and issue validation.

Trust Assumptions

There are multiple privileged roles in the vault and withdrawer system such as the PROCESSOR_ROLE, PAUSER_ROLE/UNPAUSER_ROLE, PROCESSOR_MANAGER_ROLE, ASSET_MANAGER_ROLE and more. We assume these privileged roles are not malicious.

Severity Classification

Severity classification in this report uses Immunefi Vulnerability Severity Classification System - v2.3 for smart contracts.

Critical Issues

No critical issues found.

High Issues

H01 Wrong decimal used for processAccounting in old BaseVaults

The old BaseVault has a processAccounting function to calculate all asset values in base asset and update the storage with total value. It does this by looping through all assets, get the individual asset rate from the Provider and calculate the value in Base asset as `balance * rate / baseAssetUnit` in line 612.

However this calculation is incorrect, according to the team, the rate returned by provider is denominated in base asset unit, thus the value should be `balance * rate / assetUnit`.

This does not immediately pose a threat to deployed vaults as long as their base asset and other assets share the same decimals, however admins should not add new assets with different decimals to existing vaults.

This value is fixed in the new vault lib in this audit.

Recommendation: Fix the calculation of processAccounting using individual asset's decimals instead of base asset's decimals.

Severity mapped to level 5 - Critical - Direct theft of any user funds, whether at-rest or in-motion, other than unclaimed yield. Downgraded to High due to the unlikely hood of attack.

Status update: Client confirmed this has been independently identified and fixed before previous deployment, no current contracts are affected.

Medium Issues

No medium issues found.

Low Issues

L01 Lack of docstrings

Some of the files in the codebase lack docstrings, such as

- VaultLib.sol
- AsyncWithdrawalLib.sol
- Withdrawer.sol
- OriginWithdrawalLib.sol

Recommendation: Consider adding proper docstrings to these contracts.

Status update: Fixed in commit 29df2d0bd706b4bad9ae7d784f504245cab6bb2f

L02 Possible griefing attack to stop admin deleting assets

The Vault library has a function deleteAsset to delete an existing asset if it is not the base asset and if the vault balance of the asset is 0.

However, an attacker could grief attack the vault by front running and donating a small amount of the deleting asset, this will cause the deleting asset transaction to revert. Although the attack itself does not benefit the attacker immediately, it might be used as a part of a larger attack vector.

Recommendation: Depending on the design, consider allowing deleting assets with a balance and offering ways to withdraw left over balance.

Status update: Acknowledged, won't fix. Client stated it is a low priority for them at the moment.

Notes

N01 Note in code

The AsyncWithdrawalLib.sol contains a NOTE in line 34 stating the code needs to be updated when Eigen Layer slashing is implemented.

Consider removing Notes and Todos from production codebase or do not deploy codebase that's not ready for production.

Status update: Acknowledged, won't fix. Client stated they will leave it as such and deal with it later.

N02 Unused imports

There are some unused imports in the codebase, such as

- AsyncWithdrawalLib in the BaseWithdrawer.sol
- IVault in AsyncWithdrawalLib.sol

Consider removing these unused imports.

Status update: Fixed in commit 29df2d0bd706b4bad9ae7d784f504245cab6bb2f

N03 Lack of indexed event parameter

The following 3 events in OriginWithdrawalLib.sol have no indexed parameters,

- WOETHWithdrawalRequested
- WOETHWithdrawalsClaimed
- OETHWithdrawalRequested

Consider indexing event parameters for offchain filtering.

Status update: Fixed in commit 29df2d0bd706b4bad9ae7d784f504245cab6bb2f

N04 Unused named return variables

Some functions in Withdrawer.sol use named return variables but these variables are not used in the function body. Named return variables are meant to be used in function body as an alternative to in-line return statement. For example

- The amounts return variable in the `claimWithdrawalsWOETH` function
- The totalAmount return variable in the `claimWithdrawalsWOETH` function

Consider either using or removing any unused named return variables.

Status update: Fixed in commit 29df2d0bd706b4bad9ae7d784f504245cab6bb2f