Vesper Pools delta for April ‘22

Smart Contract Security Assessment

Apr 8, 2022

ABSTRACT

Dedaub was commissioned to perform a security audit of several smart contract modules of three independent Vesper protocols.

This report focuses exclusively on the recent changes (on multiple files) in the Vesper Pools V3 protocol, in the repository https://github.com/bloqpriv/vesper-pools-v3, up to commit 0a2a497a0974a21ddb2dce51d6e6afaf7586fa87 (from commit 3f3161b037f727572a107076620e6b26ed3fb3eb). We have audited the project previously, up to the listed earlier commit.

The main change of this audit was the change in the fee logic. These included the removal of the interest and withdrawal fees, and the introduction of the universal fee which is calculated and collected upon every rebalance of a strategy.

Setting and Caveats

Our earlier audits describe the setting and caveats for Vesper Pools V3. As a general warning, we note that an audit of small changes in a large protocol is necessarily out-of-context. We made a best-effort attempt to understand the changed lines of code and assess whether these changes are reasonable and do not introduce vulnerabilities. The audit, however, was restricted to the modified lines, and their interaction with the rest of the protocol is not always easy to assess.

The audit's main target is security threats, i.e., what the community understanding would likely call "hacking", rather than regular use of the protocol. Functional

correctness (i.e., issues in “regular use“) is a secondary consideration. Typically it can only be covered if we are provided with unambiguous (i.e., full-detail) specifications of what is the expected, correct behavior. In terms of functional correctness, we often trusted the code’s calculations and interactions, in the absence of any other specification. Functional correctness relative to low-level calculations (including units, scaling, quantities returned from external protocols) is generally most effectively done through thorough testing rather than human auditing.

VULNERABILITIES & FUNCTIONAL ISSUES

This section details issues that affect the functionality of the contract. Dedaub generally categorizes issues according to the following severities, but may also take other considerations into account such as impact or difficulty in exploitation:

Category	Description
CRITICAL	Can be profitably exploited by any knowledgeable third party attacker to drain a portion of the system's or users' funds OR the contract does not function as intended and severe loss of funds may result.
HIGH	Third party attackers or faulty functionality may block the system or cause the system or users to lose funds. Important system invariants can be violated.
MEDIUM	Examples: -User or system funds can be lost when third party systems misbehave. -DoS, under specific conditions. -Part of the functionality becomes unusable due to programming error.
LOW	Examples: -Breaking important system invariants, but without apparent consequences. -Buggy functionality for trusted users where a workaround exists. -Security issues which may manifest when the system evolves.

Issue resolution includes "dismissed", by the client, or "resolved", per the auditors.

CRITICAL SEVERITY:

[No critical severity issues]

HIGH SEVERITY:

[No high severity issues]

MEDIUM SEVERITY:

[No medium severity issues]

LOW SEVERITY:

ID	Description	STATUS
L1	Inconsistent profits reported by various Earn strategies	**OPEN**
The recent change in the fee logic has required special handling of the fee in the different Earn strategies. In short, for Earn strategies the universal fee is handled by the strategy, rather than the Pool. In most cases this is achieved by having the `_realizeProfit()` method of the Strategies return 0. However this is not the case for the EarnAaveMakerStrategy and the EarnCompoundMakerStrategy strategies, which, after distributing the universal fee, return the balance in collateral tokens as their profit. This can lead the strategies to pay more fees than they should be.		

OTHER/ ADVISORY ISSUES:

This section details issues that are not thought to directly affect the functionality of the project, but we recommend considering them.

ID	Description	STATUS
A1	View function can revert unexpectedly	**OPEN**

The `currentBorrowRatio()` method of the AaveLeverageStrategy and CompoundLeverageStrategy strategies can unexpectedly revert if _supply (strategy's balance in cToken/aToken) is 0.

```
/**
* @notice Current borrow ratio, calculated as current borrow divide by max
allowed borrow
* Return value is based on basis points, i.e. 7500 = 75% ratio
*/
function currentBorrowRatio() external view returns (uint256) {
  (uint256 _supply, uint256 _borrow) = getPosition();
  return _borrow == 0 ? 0 : (_borrow * MAX_BPS) / _supply;
}
```

We believe it should return a special value instead of reverting in this case.

A2	Inconsistent coding patterns in the fee handling of Earn strategies	**OPEN**

The `_rebalanceDaiInLender()` method of the EarnVesperMakerStrategy contract deviates from the new blueprint of `_realizeProfit` by essentially including the typical `_handleProfit()` logic in the `_rebalanceDaiInLender()` method and omitting the former. The logic looks fine, but we are not confident if there is a reason why this is done this way.

A3	Compiler bugs	**INFO**

Vesper Pools contracts were compiled with the Solidity compiler v0.8.9 which, at the time of writing, has no known issues.

DISCLAIMER

The audited contracts have been analyzed using automated techniques and extensive human inspection in accordance with state-of-the-art practices as of the date of this report. The audit makes no statements or warranties on the security of the code. On its own, it cannot be considered a sufficient assessment of the correctness of the contract. While we have conducted an analysis to the best of our ability, it is our recommendation for high-value contracts to commission several independent audits, as well as a public bug bounty program.

ABOUT DEDAUB

Dedaub offers technology and auditing services for smart contract security. The founders, Neville Grech and Yannis Smaragdakis, are top researchers in program analysis. Dedaub's smart contract technology is demonstrated in the contract-library.com service, which decompiles and performs security analyses on the full Ethereum blockchain.