Vesper Synth delta for May ‘22

Smart Contract Security Assessment

June 3, 2022

ABSTRACT

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

This report focuses exclusively on the recent changes (on multiple files) in the Vesper Synth protocol, in the repository https://github.com/bloqpriv/vesper-synth, up to commit 9e084571a7c1216e28692387e47dd28a91c6914a (from commit 6ad5f509eaef48600cf15620d347cf489bbf6da1). We have audited the project previously, up to the listed earlier commit.

Setting and Caveats

Our earlier audit reports describe the setting and caveats for Vesper Synth. 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.

The delta contains only very minor changes, so no important issues were found.

CRITICAL SEVERITY:

[No critical severity issues]

HIGH SEVERITY:

[No medium severity issues]

MEDIUM SEVERITY:

[No medium severity issues]

LOW SEVERITY:

<table>
<tr><th>ID</th><th>Description</th><th>STATUS</th></tr>
<tr><td>L1</td><td>Oracles should be replaced by one-oracle</td><td>INFO</td></tr>
<tr><td colspan="3">Some of the Oracles in Vesper Synth use older code compared to one-oracle. For instance UniswapV2LikePriceProvider and UniswapV3PriceProvider contain no checks that the stable coin is pegged. Our understanding is that all oracle-related code will be replaced by one-oracle in the near future, we just mention this here to keep track of this change.</td></tr>
</table>

OTHER/ ADVISORY ISSUES:

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

<table>
<tr><th>ID</th><th>Description</th><th>STATUS</th></tr>
<tr><td>A1</td><td>Use of block.timestamp</td><td><b>INFO</b></td></tr>
<tr><td colspan="3">In the latest delta, block.timestamp is used instead of block.number to compute rewards and interest rates, and it was requested to provide feedback with respect to this change. We did not find anything problematic with it, we should just point out that block.timestamp can be manipulated to some small extent (up to about 15 seconds) by a miner, so this fact should be kept in mind for future uses of it.</td></tr>
<tr><td>A2</td><td>Compiler bugs</td><td><b>INFO</b></td></tr>
<tr><td colspan="3">Vesper Pools contracts were compiled with the Solidity compiler v0.8.9 which, at the time of writing, has two known issues:
● Nested calldata arrays are not correctly bounds checked if used in another external call or inside abi.encode().
● Conflicting stores (memory and calldata) of function arguments used in overridden functions could produce invalid code.
We have reviewed the issues and do not believe them to affect the codebase.</td></tr>
</table>

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.