Vesper

Smart Contract Audit

Nov 2021

coinspect

vesper-pools

Smart Contract Audit

V21112

Prepared for Vesper • November 2021

© 2021 Coinspect

1. Executive Summary

In **November 2021, Vesper** engaged Coinspect to perform a source code review of **vesper-pools** and **vesper-pools-v3**. The objective of the project was to continue to evaluate the security of the smart contracts.

The audit of Vesper V2 involved changes in the quorum of the `GovernorAlpha` contract, accepting multiple `vTokens` in the `PaymentSplitter` contract, removing the minimum delay in the `TimeLock` contract and fixing a known underflow issue.

Regarding Vesper V3, the changes included fixing known issues, updating the pool rewards, small changes in the `VFRStablePool`, new earn strategies and a new flash loan helper contract.

No issues were identified during the assessment:

High Risk	Medium Risk	Low Risk
0	0	0
Fixed	Fixed	Fixed
-	-	-

© 2021 Coinspect

2. Assessment and Scope

The audit started on November 2, 2021 on the repositories

- https://github.com/vesperfi/vesper-pools for the V2 version as of commit `1060ffb53864208d3d89fc7c6cd03d0fed0c4740` of **October 26, 2021**
- https://github.com/bloqpriv/vesper-pools-v3 for the V3 version as of commit `71edb604135c444b7fd46fc0adf558be0b860067` of **November 2, 2021**.

The modifications performed to the following files in Vesper V2's repository were reviewed during this engagement:

```
3a08e4e0dd51a12722cb2c1dc2d65135ff21f08f514bdd5316832d26c59c492e  Timelock.sol
7c0de8359bc095df50c7f3b549983b087c7213ad149472cca7094dfeb72f4054  governor/GovernorAlpha.sol
4b642dab2247033ab20557cda3d3f1169797e1f25b03da4909e41e7fcc7bc64e  strategies/CompoundStrategy.sol
a01b6e8b3dccb8285b1efe9fbe2128951d912c8be293c06c8fd2c3d77765e8c8  strategies/OraclesBase.sol
9ff0065718601834c5e67d46599496b692bfa92904110103f20b8a7f0c72e45d  strategies/PaymentSplitter.sol
007b7c51d5ba1dcd237ba7a2c3eb67e48989bf7a8dcf5bc1f7463f419038fafb  strategies/VSPStrategy.sol
bd658a4e9f824b1af2d3967fdf6fac30bedbfffce355bdb8e9b24978a7c3472e  strategies/VesperV3Strategy.sol
```

The modifications performed to the following files in Vesper V3's repository were reviewed during this engagement:

```
ffaf0f4f59b48fc1c15cde284f7af6d8a953f01f441b29cdaecc0d69906ca980  FlashLoanHelper.sol
8d47375cac78cc74f4f4a55279d2c0a2071fb4d36caa78d1c298f44d080c556c  interfaces/aave/IAave.sol
c3983d48bc1eb6ce5472a201deeec8356181b34e43a0aad3e7022364db68c642  interfaces/bloq/IAddressList.sol
a0286d7f0ec74a4d093607fdd62068bbf21d86a579ef9f8904f54e93da351952  interfaces/convex/IConvexToken.sol
47dd50fdc754b10970a368c5152f577b42fe3645893d97c68547c63909de634a  interfaces/dydx/ISoloMargin.sol
437838717955aef24b6b5612a9a4f610c1fde2069c3dd81ac837f94052a74fc5  interfaces/oracle/IUniswapV3Oracle.sol
3e8799bd3d3111f8c72dfd318a8bf8e282d360c824d3ceea19389aa50a4e66da  interfaces/vesper/IStrategy.sol
fea09c462d555c06f5f5be5ab8b3948e4456711702dbe7783f0fa67b544d7f6f  pool/PoolAccountant.sol
13708f0c84d17bced3065b3edc646be635c8768a10d696fb71c63c6be6abe72b  pool/PoolRewards.sol
3f8752c421e6e64db53482c84f6da9ded5df88e0925770d858ff5f7f1df87470  pool/PoolShareToken.sol
f2bdbf5796b9b6a5ae0cbbf516b9dd1913a55c917fcc043a0c28c1b8892b3492  pool/VPoolBase.sol
247cabcdbb29e78bfa3c57f1baca879b2308f0169e0dfefa93bfc9fe4f3ac37b  pool/earn/EarnDrip.sol
aea64e4d840b271f00523612561b6d3297738bd2d0b1aff65e7379a3f3a9807f  pool/vfr/VFRStablePool.sol
623eb66d0cbd84ffdf10504bc9ab384f221e7d9fc1a13f3b1e07edaaeff99e1f  strategies/Earn.sol
7e6b57f301621e08e8fe8e058b80ab48fd626c408d4ba31133e1f2a62b01489c  strategies/Strategy.sol
1138bd58923528ec3de9d019ccb9f4897d33d61847426fa8ea1c86ec27b244e5  strategies/VFR.sol
0c371a029f40217edd7d63e73222a796af9fad5514435b05d52304897bd544a3  strategies/aave/AaveStrategyPolygon.sol
afb7cf8387d409e7c93dd97673ac4df6fff960170d7367b9128ca330e3ff20ee  strategies/compound/CompoundLeverageStrategy.sol
0c2f2d2631ee61438d4d4ba252d652d8fcd870a320a30b49ccbbad3278306035  strategies/compound/CompoundLeverageStrategyUNI.sol
34b64a60c26f7fd5a13144168a52540b9648ac93992dbdd9ffaf6fca9c55b54c  strategies/compound/CompoundStrategyWBTC.sol
b5e3099ebde18cc94a0009a673471e914a3a5ba74ff8045f4a734fca1f224415  strategies/compound/CompoundXYStrategy.sol
1c0d6432edc8711910fc152fd855dffa58a8a57b052857fbabaf119d3b7abe45  strategies/compound/earn/EarnCompoundStrategyWBTC.sol
098b3d45077a3d40a7954f69891eaf14a25068a6e4bf36894180cf938c8fd177  strategies/convex/ConvexSBTCStrategyWBTC.sol
3e1478ac8ac59c167b922c76d2ebae61f20f37af0794675a87f0a55485af15df  strategies/convex/ConvexStrategy.sol
e23fbc200fdc4dfd467bee90dade8dd7b711bf48b1985e5e5de334eb75bd6f53  strategies/curve/3Pool/CrvsBTCPoolStrategy.sol
e748da050def3bfebd934c7073b9115fc2480447b4b3549238df622c50c2df69  strategies/curve/3Pool/earn/EarnCrvsBTCStrategy.sol
a81aa1a7e14fb4256f2a74d0f75a4984871d210d8563af8a26d6747cbe0bcf31  strategies/curve/3Pool/earn/EarnCrvsBTCStrategyWBTC.sol
89deffddddc1bdc0ccaa5c7db5bd18ebc4301eaca9220c5175d97fec7c72bd0b  strategies/curve/Crv3PoolStrategyBase.sol
f85414a10888fa4498399c9e491563de84f617d47f89c276e064c9bd4a6c8b8d  strategies/maker/MakerStrategy.sol
3afefaa2a8749c94a34fbf0bfa082e99789c9dc6e2f93803d6eb0b1a743c49db  strategies/maker/VesperMakerStrategy.sol
d9ff4b1d20779d8a771d3162eea7adc67fbcf2e2dc9fc12b088a4d2485285f60  strategies/maker/VesperMakerStrategyETH.sol
3364c458d4fc0b2e6016cfacd2894a95001c4c4e40df5013279be79dc9d586ca  strategies/maker/VesperMakerStrategyWBTC.sol
bfceef58736be510dc5c0a6fde8d4a5b49cbd9c46f4e70046f6b0a925225d67e  strategies/maker/earn/EarnAaveMakerStrategy.sol
6dd72670f7179db6f30f64f45431b8146ff1a713d444f38ac3782003db7e7dea  strategies/maker/earn/EarnAaveMakerStrategyETH.sol
```

```
5159891a559afa5ee107fa6ce5ff435cb2739b08a5b540fab039307da1a21e5a  strategies/maker/earn/EarnCompoundMakerStrategy.sol
67a05851cfaf877c4d7ebc19eef30bc2e5c4a61bc9f422b5810e873f1c64bbc9  strategies/maker/earn/EarnCompoundMakerStrategyETH.sol
1b0bb341f643791511e08217b8c4ef957f8af86fbd7ba22d417bc095ebf8beac  strategies/maker/earn/EarnVesperMakerStrategy.sol
909820f2fc2d662075eb50d79b4dc7446f3de7bc650f070830361d3c7809b8cf  strategies/maker/earn/EarnVesperMakerStrategyETH.sol
397f652e96109e64d0764937d5c49f731481c7e949a9fe3acc085c023f33940a  strategies/maker/earn/EarnVesperMakerStrategyWBTC.sol
86aac3b79122d56572a25d42ace1422885ec9532b0f1461617c2550a0beb9136  strategies/vesper/EarnVesperStrategy.sol
11b08b250ff1a2178daf0766ee2b32584343da2294123e333904795aa649b211  strategies/vesper/EarnVesperStrategyDAIWBTC.sol
22eb3d44dd4bb54d984e6280a2f0a38ce7cbba897d7ae6628ebafd4f9bbe5507  strategies/vesper/EarnVesperStrategyDAIWETH.sol
3466c2d81624e81b02913e7b0e9562a700dc10e7a8d64aa7cfe529ea68f55a29  strategies/yearn/earn/EarnYearnStrategy.sol
11e86406548c78496e9aad955e3aa631de5c54c554c14a2b398edcea86e01da5  upgraders/PoolRewardsUpgrader.sol
```

Vesper V2 changes

The quorum updates in the `GovernorAlpha` added a requirement of a minimum of 25% positive votes for the approval of a proposal.

This modification creates potentially undesired incentives in the protocol because:

- Those who will vote for the proposal might want to wait until the last minute to cast their vote. This is due to the fact that if some opponents of the proposal are waiting to see if there is enough quorum to cast their vote, last minute voting might pass under the radar.
- As a consequence, those against the proposal are incentivized to vote early on.

In this new scenario, if 25% of the voters secretly form a coalition, they can attack the protocol by submitting 25 times the same proposal forcing all opponents to vote early or risk that at least one of those proposals might pass. This would make it more expensive to be against a proposal than to be in favor of it.
A small change that would prevent this situation is to extend the voting period once the proposal reaches the quorum if the time left is below a certain threshold. The proposed improvement would have the extra benefit of allowing negative votes to be cast only on proposals that have a quorum, saving on transaction costs for all the users of the platform on average.

The underflow issue in `CompoundStrategy` was correctly fixed. It is not possible anymore to revert the `_calculatePendingFee` call.

The `PaymentSplitter` contract now supports multiple `vTokens` for topping it up. The function has no security issues. There are two unbounded loops that may lead

to denial of service problems, but require unlikely errors done by the administrator. In particular in the `_topUp` and `removeVToken` functions.

The lower bound delay on the `TimeLock` contract has been removed. Now there is no minimum delay, but there is still a maximum delay that can be set.

Vesper V3 changes

As stated in the documentation, the `FlashLoanHelper` contract does all the heavy lifting to get flash loans via Aave and DyDx. It provides internal functions so that another contract can inherit from it and provide flash loans for end users. The `FlashLoanHelper` is currently used by the `CompoundLeverageStrategy`.

There are two minor observations about the helper. The first one is that the `_approveToken` function approves both lending platforms when it would be better for them to be approved separately. The other observation is that the `awaitingFlash` variable usage might be ineffective. It is possible to cause a reentrancy on the `executeOperation` function without passing through the `_doAaveFlashLoan` again as this variable is already set to true. Coinspect recommends setting `awaitingFlash` to `false` immediately after checking its value in `executeOperation`. Coinspect auditors did not find a concrete exploitation path, but it may produce errors in the future.

The `CompoundLeverageStrategy` also changed the borrow ratio logic. In this regard, there is a `require` statement that might not hold over time in the `updateBorrowRatio` function:

```
function updateBorrowRatio(...)
      (, uint256 _collateralFactor, ) = COMPTROLLER.markets(address(cToken));
      require(_maxBorrowRatio < (_collateralFactor / 1e14), "invalid-max-borrow-limit")
```

The collateral factor is a value that might change at any time resulting in the `maxBorrowRatio` being off limits suddenly.

Changes in the `VFRStablePool` include a minimum lock period and an `autoRetarget` method. The lock period works by saving the deposit timestamp and comparing the current block timestamp with the deposit time plus the lock period.

As a consequence, the lock period can be skipped by the governor by setting the lock period to 0 and recovering the normal period after exploitation. However, protocol users are subject to any new locking period with immediate application on deposits done while the old lock period was ruling. Coinspect suggests evaluating an alternative approach: instead of saving the deposit timestamp, the map can save the withdrawal timestamp estimated at deposit time. The deposit function could have the following code:

```
withdrawalTimestamp[_msgSender()] = max(
    withdrawalTimestamp[_msgSender()],
    block.timestamp + lockPeriod
)
```

Another minor issue in the `VFRStablePool` is that the `autoRetarget` function might underflow when `targetAPY < tolerance`. This unlikely situation might even happen with successives calls to the `autoRetarget` function.

The `CompoundXYStrategy` now incorporates a `recoverBorrowToken` function that allows to transfer idle borrow tokens to the pool.

The last set of changes are the new `Earn` strategies. Coinspect verified the new logic implemented, the usages of external APIs and the addresses of the third party contracts used. Coinspect found no issues regarding these strategies.

The following third party addresses utilized in the new contracts reviewed were verified to be correct:

1. `0xB53C1a33016B2DC2fF3653530bfF1848a515c8c5` for `AAVE` Lending Pool Addresses Provider as shown in https://docs.aave.com/developers/deployed-contracts/deployed-contracts
2. `0xC02aaA39b223FE8D0A0e5C4F27eAD9083C756Cc2` for `WETH9` as shown in https://etherscan.io/address/0xc02aaa39b223fe8d0a0e5c4f27ead9083c756cc2#code
3. `0x4Ddc2D193948926D02f9B1fE9e1daa0718270ED5` for `cETH` as listed in https://compound.finance/docs

4. `0x6B175474E89094C44Da98b954EedeAC495271d0F` for `DAI` as listed in https://github.com/makerdao/developerguides/blob/master/dai/dai-token/dai-token.md
5. `0x028171bCA77440897B824Ca71D1c56caC55b68A3` for `aDAI Token v2` as listed in https://docs.aave.com/developers/deployed-contracts/deployed-contracts
6. `0x35A18000230DA775CAc24873d00Ff85BccdeD550` for `cUNI` as listed in https://compound.finance/docs
7. `0xccF4429DB6322D5C611ee964527D42E5d685DD6a` for `cWBTC2` as listed in https://compound.finance/docs
8. `0x2260FAC5E5542a773Aa44fBCfeDf7C193bc2C599` for `WBTC` as listed in https://etherscan.io/address/0x2260fac5e5542a773aa44fbcfedf7c193bc2c599#code
9. `0x1E0447b19BB6EcFdAe1e4AE1694b0C3659614e4e` for `DyDx SoloMargin` as listed in https://etherscan.io/address/0x1e0447b19bb6ecfdae1e4ae1694b0c3659614e4e#code
10. `0x0F1f5A87f99f0918e6C81F16E59F3518698221Ff` for `Cross Pool Oracle` as listed in https://etherscan.io/address/0x0F1f5A87f99f0918e6C81F16E59F3518698221Ff#code and https://andrecronje.medium.com/easy-on-chain-oracles-54d82961a2a0

It is worth noting that some of these contracts proxy the implementation contracts and/or are upgradable and not in control of Vesper's team.

3. Disclaimer

The information presented in this document is provided "as is" and without warranty. The present security audit does not cover any off-chain systems or frontends that communicate with the contracts, nor the general operational security of the organization that developed the code.