Cetus Vault Smart Contract

Audit Report

contact@movebit.xyz

https://twitter.com/movebit_

Thu Feb 29 2024

Cetus Vault Smart Contract Audit Report

1 Executive Summary

1.1 Project Information

Description	Cetus is a pioneer DEX and concentrated liquidity protocol built on the Sui and Aptos blockchain.
Type	DeFi
Auditors	MoveBit
Timeline	Fri Feb 23 2024 - Tue Feb 27 2024
Languages	Move
Platform	Sui
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/CetusProtocol/vaults/
Commits	464c943caf624e3434986280b4a02e275e6a8bba c23ab3489af813cad58a4cdfbad0f2600a40d06a

1.2 Files in Scope

The following are the SHA1 hashes of the original reviewed files.

ID	File	SHA-1 Hash
FET	sources/fetcher.move	9905c7eeb41f86e2dac0969b0a8509dbfc8d3092
MOV	Move.toml	d3be829340101fe9166fb9ad76b91f6ede072293
ACL	sources/acl.move	62526f6fa29e4adcbc1b65f0d58bf7113164d6bc
ROU	sources/router.move	1d38a14f6b123042007521534f70f6e65c37426f
UTI	sources/utils.move	0214fe9d0f26d058af4273e8bed364573406f7e3
VAU	sources/vaults.move	23a58e983a5725c0a7b9b8c0346d39e27f7e4422

1.3 Issue Statistic

Item	Count	Fixed	Acknowledged
Total	4	2	2
Informational	0	0	0
Minor	0	0	0
Medium	3	2	1
Major	1	0	1
Critical	0	0	0

1.4 MoveBit Audit Breakdown

MoveBit aims to assess repositories for security-related issues, code quality, and compliance with specifications and best practices. Possible issues our team looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Integer overflow/underflow by bit operations
- Number of rounding errors
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic contradicting the specification
- Code clones, functionality duplication
- Gas usage
- Arbitrary token minting
- Unchecked CALL Return Values
- The flow of capability
- Witness Type

1.5 Methodology

The security team adopted the **"Testing and Automated Analysis"**, **"Code Review"** and **"Formal Verification"** strategy to perform a complete security test on the code in a way that is closest to the real attack. The main entrance and scope of security testing are stated in the conventions in the "Audit Objective", which can expand to contexts beyond the scope according to the actual testing needs. The main types of this security audit include:

(1) Testing and Automated Analysis

Items to check: state consistency / failure rollback / unit testing / value overflows / parameter verification / unhandled errors / boundary checking / coding specifications.

(2) Code Review

The code scope is illustrated in section 1.2.

(3) Formal Verification

Perform formal verification for key functions with the Move Prover.

(4) Audit Process

- Carry out relevant security tests on the testnet or the mainnet;
- If there are any questions during the audit process, communicate with the code owner in time. The code owners should actively cooperate (this might include providing the latest stable source code, relevant deployment scripts or methods, transaction signature scripts, exchange docking schemes, etc.);
- The necessary information during the audit process will be well documented for both the audit team and the code owner in a timely manner.

2 Summary

This report has been commissioned by Cetus to identify any potential issues and vulnerabilities in the source code of the Cetus Vault smart contract, as well as any contract dependencies that were not part of an officially recognized library. In this audit, we have utilized various techniques, including manual code review and static analysis, to identify potential vulnerabilities and security issues.

During the audit, we identified 4 issues of varying severity, listed below.

ID	Title	Severity	Status
ROU-1	User Rewards Lost	Medium	Acknowledged
VAU-1	Centralization Risk	Major	Acknowledged
VAU-2	Missing `vault.status` Check	Medium	Fixed
VAU-3	Unprocessed Variable	Medium	Fixed

3 Participant Process

Here are the relevant actors with their respective abilities within the Cetus Vault Smart Contract :

Admin

- The Admin can set roles for a member through `set_roles()` .
- The Admin can add a role for a member through `add_role()` .
- The Admin can remove a role for a member through `remove_role()` .
- The Admin can remove a member from `ACL` through `remove_member()` .
- The Admin can update the package version through `update_package_version()` .
- The Admin can add oracle pool of pair `<CoinTypeA, CoinTypeB>` into `VaultManager` through `add_oracle_pool()` .
- The Admin can remove oracle pool of pair `<CoinTypeA, CoinTypeB>` from `VaultManager` through `remove_oracle_pool()` .
- The Admin can update `<CoinTyppeA, CoinTypeB>` pair oracle pool `slippage` through `update_slippage()` .
- The Admin can create a `Vault` through `create_vault<CoinTypeA, CoinTypeB, T>()` .
- The Admin can update the protocol fee rate through `update_protocol_fee_rate<T>()` .
- The Admin can update vault `max_quota` through `update_max_quota<T>()` .
- The Admin can update vault `finish_rebalance_threshold` through `update_rebalance_thresold<T>()` .
- The Admin can flash loan/repay the flash loan from harvest assets of the vault through `flash_loan<CoinTypeA, CoinTypeB, T>()/repay_flash_loan<CoinType, T>()` .
- The Admin can reinvest the `CoinA` and `CoinB` in `Vault` through `reinvest<CoinTypeA, CoinTypeB, T>()` .
- The Admin can migrate the liquidity from the old pool to a new pool through `migrate_liquidity<CoinTypeA, CoinTypeB, T>()` .
- The Admin can rebalance the `Vault` through `rebalance<CoinTypeA, CoinTypeB, T>()` .

- The Admin can collect rewarders of `Vault` through `collect_rewarder<CoinTypeA, CoinTypeB, CoinTypeC, T>`.
- The Admin can harvest farm rewarders through `harvest<RewardCoin, T>()`.
- The Admin can claim the protocol fee by `CoinType` through `claim_protocol_fee<CoinType, T>()`.
- The Admin can pause the `Vault` through `pause<T>()`.
- The Admin can unpause the `Vault` through `unpause<T>()`.

User

- The User can deposit tokens to `Vault` and get the `Lp` token through `deposit<CoinTypeA, CoinTypeB, T>()`.
- The User can burn the `Lp` token and get the deposit token with rewards through `remove<CoinTypeA, CoinTypeB, T>()`.

4 Findings

ROU-1 User Rewards Lost

Severity: Medium

Status: Acknowledged

Code Location:

sources/router.move#154

Descriptions:

In the `remove` function, rewards are not updated and reinvested back into the pool. Before a user removes `LP` tokens from the `Vault`, rewards obtained by the `Vault` should be updated and reinvested back into the `pool` to ensure that users receive their rightful profits.

Suggestion:

It is recommended to confirm if it aligns with the design.

Resolution:

The client replied that the off-chain robot will reinvest once the income reaches a certain amount and at least once a day. When the users remove liquidity, the income generated since the last reinvestment is minimal, so there will be no reinvest when users remove liquidity.

VAU-1 Centralization Risk

Severity: Major

Status: Acknowledged

Code Location:

sources/vaults.move

Descriptions:

Centralization risk was identified in the smart contract.

- The Admin can set roles for a member through `set_roles()`.
- The Admin can add a role for a member through `add_role()`.
- The Admin can remove a role for a member through `remove_role()`.
- The Admin can remove a member from `ACL` through `remove_member()`.
- The Admin can update the package version through `update_package_version()`.
- The Admin can add oracle pool of pair `<CoinTypeA, CoinTypeB>` into `VaultManager` through `add_oracle_pool()`.
- The Admin can remove oracle pool of pair `<CoinTypeA, CoinTypeB>` from `VaultManager` through `remove_oracle_pool()`.
- The Admin can update `<CoinTyppeA, CoinTypeB>` pair oracle pool `slippage` through `update_slippage()`.
- The Admin can create a `Vault` through `create_vault<CoinTypeA, CoinTypeB, T>()`.
- The Admin can update the protocol fee rate through `update_protocol_fee_rate<T>()`.
- The Admin can update vault `max_quota` through `update_max_quota<T>()`.
- The Admin can update vault `finish_rebalance_threshold` through `update_rebalance_thresold<T>()`.
- The Admin can flash loan/repay the flash loan from harvest assets of the vault through `flash_loan<CoinTypeA, CoinTypeB, T>()/repay_flash_loan<CoinType, T>()`.
- The Admin can reinvest the `CoinA` and `CoinB` in `Vault` through `reinvest<CoinTypeA, CoinTypeB, T>()`.

- The Admin can migrate the liquidity from the old pool to a new pool through `migrate_liquidity<CoinTypeA, CoinTypeB, T>()`.
- The Admin can rebalance the `Vault` through `rebalance<CoinTypeA, CoinTypeB, T>()`.
- The Admin can collect rewarders of `Vault` through `collect_rewarder<CoinTypeA, CoinTypeB, CoinTypeC, T>`.
- The Admin can harvest farm rewarders through `harvest<RewardCoin, T>()`.
- The Admin can claim the protocol fee by `CoinType` through `claim_protocol_fee<CoinType, T>()`.
- The Admin can pause the `Vault` through `pause<T>()`.
- The Admin can unpause the `Vault` through `unpause<T>()`. Any potential leaks or malicious manipulation could lead to serious issues.

Suggestion:

It is recommended to confirm if it aligns with the design.

Resolution:

The client replied that the Admin account will be set as a multi-signature account for management and protection.

VAU-2 Missing `vault.status` Check

Severity: Medium

Status: Fixed

Code Location:

sources/vaults.move#896,963,1226,1260,1298

Descriptions:

In the `migrate_liquidity` function, the status of the variable `vault.status` will be changed to `STATUS_REBALANCING`. During this status, execution of other functions such as `flash_loan`, `repay_flash_loan`, `collect_fee`, `harvest`, etc., should be avoided.

Suggestion:

It is recommended to confirm if it aligns with the design.

Resolution:

This issue has been fixed. The client has adopted our suggestions.

VAU-3 Unprocessed Variable

Severity: Medium

Status: Fixed

Code Location:

sources/vaults.move#153

Descriptions:

The variable `flash_loan_count` in the `Vault` struct is not modified during the flash loan, and it is also not modified by other functions in the contract.

Suggestion:

It is recommended to confirm if it aligns with the design.

Resolution:

This issue has been fixed. The client has adopted our suggestions.

Appendix 1

Issue Level

- **Informational** issues are often recommendations to improve the style of the code or to optimize code that does not affect the overall functionality.
- **Minor** issues are general suggestions relevant to best practices and readability. They don't post any direct risk. Developers are encouraged to fix them.
- **Medium** issues are non-exploitable problems and not security vulnerabilities. They should be fixed unless there is a specific reason not to.
- **Major** issues are security vulnerabilities. They put a portion of users' sensitive information at risk, and often are not directly exploitable. All major issues should be fixed.
- **Critical** issues are directly exploitable security vulnerabilities. They put users' sensitive information at risk. All critical issues should be fixed.

Issue Status

- **Fixed:** The issue has been resolved.
- **Partially Fixed:** The issue has been partially resolved.
- **Acknowledged:** The issue has been acknowledged by the code owner, and the code owner confirms it's as designed, and decides to keep it.

Appendix 2

Disclaimer

This report is based on the scope of materials and documents provided, with a limited review at the time provided. Results may not be complete and do not include all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your own risk. A report does not imply an endorsement of any particular project or team, nor does it guarantee its security. These reports should not be relied upon in any way by any third party, including for the purpose of making any decision to buy or sell products, services, or any other assets. TO THE FULLEST EXTENT PERMITTED BY LAW, WE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, IN CONNECTION WITH THIS REPORT, ITS CONTENT, RELATED SERVICES AND PRODUCTS, AND YOUR USE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NOT INFRINGEMENT.