Turbos Vault Smart Contract

Audit Report

MOVEBIT

contact@movebit.xyz

https://twitter.com/movebit_

Mon Apr 22 2024

Turbos Vault Smart Contract Audit Report

1 Executive Summary

1.1 Project Information

Description	An assets manage project
Type	DeFi
Auditors	MoveBit
Timeline	Thu Apr 04 2024 - Mon Apr 08 2024
Languages	Move
Platform	Sui
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/turbos-finance/liquidity-vault
Commits	edb2c6d52c3fd068be9c60e573fa322f341e1932 bdcf75f76d8dbeb849ad3189cc0743aebda08f8b dfcccaec31f0a18a6bdbdccd928e22a06db5115e 0a6d72c85c03474dcfea9fa4272fdbe45e6d3ab3

1.2 Files in Scope

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

ID	File	SHA-1 Hash
MOV	Move.toml	e9b59289eac3b232c5ada14e530448250801e430
ACL	sources/acl.move	8eb698b18798dc96c97e7df6ab89a15968b8a03e
REW	sources/rewarder.move	94350669e5730eb3cd464bb82953115a858cfd07
ROU	sources/router.move	a31b80274d747015e0dbd7fa8f357ec2884e852e
CON	sources/config.move	8a619d5d1232c83a3e51299566c8fa3515c5637e
VAU	sources/vault.move	4b676381ab9ae083903c2a7a675cecd1389df68b

1.3 Issue Statistic

Item	Count	Fixed	Acknowledged
Total	5	3	2
Informational	2	0	2
Minor	0	0	0
Medium	0	0	0
Major	3	3	0
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 Turbos to identify any potential issues and vulnerabilities in the source code of the Turbos 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 5 issues of varying severity, listed below.

ID	Title	Severity	Status
VAU-1	Unrestricted Access to The `update_reward_info` Function	Major	Fixed
VAU-2	Incorrect Reward Update	Major	Fixed
VAU-3	Missing Token Type Check	Major	Fixed
VAU-4	Unused Variable	Informational	Acknowledged
VAU-5	Unused Function	Informational	Acknowledged

3 Participant Process

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

Admin

- The Admin can set roles for the address through `set_roles()`.
- The Admin can add a role for the address through `add_role()`.
- The Admin can remove a role for the address through `remove_role()`.
- The Admin can add an operator role for the address and transfer the `OperatorCap` to it through `add_role()`.
- The Admin can set the package version through `set_package_version()`.

Operator

- The Operator can create a strategy through `create_strategy<CoinTypeA, CoinTypeB, FeeType>()`.
- The Operator can collect the protocol fee of the strategy through `collect_protocol_fee<BalanceType>()`.
- The Operator can update the strategy protocol fee rate through `update_strategy_protocol_fee_rate()`.
- The Operator can rebalance the strategy through `rebalance<CoinTypeA, CoinTypeB, FeeType>()`.
- The Operator can update the strategy status through `update_strategy_status()`.
- The Operator can update the strategy default base rebalance threshold through `update_strategy_default_base_rebalance_threshold()`.
- The Operator can update the strategy default limit rebalance threshold through `update_strategy_default_limit_rebalance_threshold()`.
- The Operator can update the strategy base minimum threshold through `update_strategy_base_minimum_threshold()`.
- The Operator can update the strategy limit minimum threshold through `update_strategy_limit_minimum_threshold()`.

- The Operator can update the vault base rebalance threshold through `update_vault_base_rebalance_threshold()`.
- The Operator can update the vault limit rebalance threshold through `update_vault_limit_rebalance_threshold()`.

User

- The User can open a vault of the strategy through `open_vault<CoinTypeA, CoinTypeB, FeeType>()`.
- The User can open a vault of the strategy and deposit in it through `open_vault_and_deposit<CoinTypeA, CoinTypeB, FeeType>()`.
- The User can close the vault through `close_vault<CoinTypeA, CoinTypeB>()`.
- The User can deposit in the vault through `deposit<CoinTypeA, CoinTypeB, FeeType>()`.
- The User can withdraw from the vault through `withdraw<CoinTypeA, CoinTypeB, FeeType>()`.
- The User can withdraw the reward from strategy through `collect_clmm_reward_direct_return<CoinTypeA, CoinTypeB, FeeType, RewardCoinType>()` and `collect_reward<RewardCoinType>()`.

4 Findings

VAU-1 Unrestricted Access to The `update_reward_info` Function

Severity: Major

Status: Fixed

Code Location:

sources/vault.move#1545

Descriptions:

The visibility of the `update_reward_info` function is `public` and there is no permission check, and the share of the vault will be reset in the `update_reward_info` function. It allows `vault_info.share` to be arbitrarily modified, leading to incorrect reward calculations.

Suggestion:

It is recommended to change the visibility of the `update_reward_info` function or add a permission check in the function.

VAU-2 Incorrect Reward Update

Severity: Major

Status: Fixed

Code Location:

sources/vault.move#1604

Descriptions:

In the `update_vault_reward_info` function, the calculate of the reward should `reward_info.reward + old_reward_share - reward_info.reward_debt`, but not `reward_info.reward + new_reward_share - reward_info.reward_debt`.

Suggestion:

It is recommended to modify the calculation of reward update.

VAU-3 Missing Token Type Check

Severity: Major

Status: Fixed

Code Location:

sources/vault.move#372,470,614,1806

Descriptions:

The administrator specifies the token type when creating a strategy, but users can operate with tokens that are not specified in the contract. This allows users to deceive liquidity rewards.

Suggestion:

It is recommended to add a `clmm pool id` check for these main functions.

VAU-4 Unused Variable

Severity: Informational

Status: Acknowledged

Code Location:

sources/vault.move#1060,1117,1248,1585,1661

Descriptions:

There are unused variables in the contract.

```
let sender = tx_context::sender(ctx);


let tick_space_i32 = i32::from_u32(tick_space);


let balance_value = balance::value<BalanceType>(&balance);
```

Suggestion:

It is recommended to remove the unused variable if there's no further design.

VAU-5 Unused Function

Severity: Informational

Status: Acknowledged

Code Location:

sources/vault.move#2035,2068,2108

Descriptions:

There are unused functions in the contract.

```
fun borrow_clmm_base_nft

fun borrow_clmm_limit_nft

public(friend) fun update_strategy_image_url
```

Suggestion:

It is recommended to remove the unused function if there's no further design.

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.