Yield Optimizer

Audit Report

MOVEBIT

contact@movebit.xyz

https://twitter.com/movebit_

Wed Nov 15 2023

Yield Optimizer Audit Report

1 Executive Summary

1.1 Project Information

Description	A yield optimizer and its published package at id 0x01c389a85310b47e7630a9361d4e71025bc35e4999d3a645949b1b68b26f2273 on Sui mainnet matches that the source code from this audit.
Type	DeFi
Auditors	MoveBit
Timeline	Mon Nov 06 2023 - Wed Nov 15 2023
Languages	Move
Platform	Sui
Methods	Architecture Review, Unit Testing, Manual Review

1.2 Files in Scope

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

ID	File	SHA-1 Hash
MOV11	yield-optimizer/Move.toml	031741f0fd900637e6cade4cc1b5124d9c3dc6ee
TLB	yield-optimizer/sources/time_locked_balance.move	d146fd6a33ab509a5d0155770328a7d16cb3314d
UTI	yield-optimizer/sources/util.move	04ccca4a793d4d03a9ae31ef29f3985950447b02
SWH	yield-optimizer/sources/scallop_whusdce.move	b6b3df6fed1e49ce62f120895b8a86588fe8a92a
VAU	yield-optimizer/sources/vault.move	e8075367f3b702650512dad3dca061fa649369da
YWH	yield-optimizer/sources/ywhusdce.move	8c0028b8fc3c1f0e827d5dc4b454eceae3046c21

1.3 Issue Statistic

Item	Count	Fixed	Acknowledged
Total	4	0	4
Informational	1	0	1
Minor	1	0	1
Medium	1	0	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 Kuna Labs to identify any potential issues and vulnerabilities in the source code of the Yield Optimizer 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
SWH-1	Unable to Remove Strategies	Medium	Acknowledged
SWH-2	Missing Emit Event	Minor	Acknowledged
VAU-1	Centralization Risk	Major	Acknowledged
YWH-1	Improper `CoinMetadata` Processing	Informational	Acknowledged

3 Participant Process

Here are the relevant actors with their respective abilities within the Yield Optimizer Smart Contract:

Admin

- The `Admin` can set the deposit threshold through `set_tvl_cap<T, YT>()`.
- The `Admin` can set the duration of profit to unlock through `set_profit_unlock_duration_sec<T, YT>()`.
- The `Admin` can set the performance fee in basis points through `set_performance_fee_bps<T, YT>()`.
- The `Admin` can withdraw the performance fee of the vault through `withdraw_performance_fee<T, YT>()`.
- The `Admin` can pull the unlocked profits to the free balance in the `vault` through `pull_unlocked_profits_to_free_balance<T, YT>()`.
- The `Admin` can join the strategies to the `vault` through `join_vault()`.
- The `Admin` can pull the coins of the strategies from `scallop_pool` through `remove_from_vault`.
- The `Admin` can remove the strategies from the `vault` through `remove_strategy<T, YT>()`.
- The `Admin` can upgrade the version of the `vault` through `migrate<T, YT>()`.
- The `Admin` can upgrade the version of the `strategies` through `migrate()`.
- The `Admin` can rebalance the strategies through `rebalance()`.
- The `Admin` can sell the profits of the strategies through `take_profits_for_selling()`.
- The `Admin` can deposit the profits to the strategies through `deposit_sold_profits()`.

User

- The `User` can deposit coins to the `vault` and get `lp` token through `deposit<T, YT>()`.
- The `User` can burn the `lp` token and get the WithdrawTicket through `withdraw<T, YT>()`.

- The `User` can withdraw their deposit coins and profits through `redeem_withdraw_ticket()<T, YT>` and `withdraw()`.

4 Findings

SWH-1 Unable to Remove Strategies

Severity: Medium

Status: Acknowledged

Code Location:

yield-optimizer/sources/scallop_whusdce.move#137;

yield-optimizer/sources/vault.move#345

Descriptions:

The `remove_from_vault` function in the `scallop_whusdce` module returns a struct named `StrategyRemovalTicket`, acting as a hot potato that can only be processed by calling the `remove_strategy` function. However, the `remove_strategy` function has the visibility set to `friend` and is not utilized in the corresponding friend module `scallop_whusdce`. As a result, the strategies in the vault will not be removed.

Suggestion:

It seems that the code implementation is incomplete. It is recommended to double-check to ensure it aligns with the design.

Resolution:

The client replied this will be fixed in a subsequent smart contract upgrade by removing the friend declaration from `remove_strategy` function.

SWH-2 Missing Emit Event

Severity: Minor

Status: Acknowledged

Code Location:

yield-optimizer/sources/scallop_whusdce.move#117,137,241,265;

yield-optimizer/sources/vault.move#224-381

Descriptions:

The smart contract lacks appropriate events for monitoring sensitive operations(such as managing assets and modifying key configs), which could make it difficult to track important actions or detect potential issues.

Suggestion:

It is recommended to emit events for these sensitive functions to make it easier to track important actions or detect potential issues.

Resolution:

The client replied it's straightforward to derive state changes from TX effects On Sui, so it's not necessary to emit events here.

VAU-1 Centralization Risk

Severity: Major

Status: Acknowledged

Code Location:

yield-optimizer/sources/vault.move;

yield-optimizer/sources/scallop_whusdce.move

Descriptions:

The `Admin` has the following privileges:

- The `Admin` can sell the profits of the strategies through `take_profits_for_selling()`.
- The `Admin` can withdraw the performance fee of the vault through `withdraw_performance_fee<T, YT>()`.
- The `Admin` can set the deposit threshold through `set_tvl_cap<T, YT>()`.
- The `Admin` can set the unlock duration through `set_profit_unlock_duration_sec<T, YT>()`.
- The `Admin` can set the performance fee in basis points through `set_performance_fee_bps<T, YT>()`.
- The `Admin` can pull the unlocked profits in the `vault` through `pull_unlocked_profits_to_free_balance<T, YT>()`.
- The `Admin` can join the strategies to the `vault` through `join_vault()`.
- The `Admin` can pull the coin of the strategies from `scallop_pool` through `remove_from_vault`.
- The `Admin` can remove the strategies from the `vault` through `remove_strategy<T, YT>()`.
- The `Admin` can upgrade the version of the `vault` through `migrate<T, YT>()`.
- The `Admin` can upgrade the version of the `strategies` through `migrate()`.
- The `Admin` can rebalance the strategies through `rebalance()`.
- The `Admin` can deposit the profits to the strategies through `deposit_sold_profits()`.

Suggestion:

It is recommended to take some measures to mitigate centralization risk.

Resolution:

The client replied this was marked due to the holder of Admin capability being able to affect the Vault profits and APR. The profits can be affected by increasing the performance fee to a high level or ineffective reward selling. This can reduce (or increase) the APR of the Vault but never affect the principal user deposited funds. If the Vault APR is unsatisfactory, the depositors can always withdraw their funds and move them elsewhere.

YWH-1 Improper `CoinMetadata` Processing

Severity: Informational

Status: Acknowledged

Code Location:

yield-optimizer/sources/ywhusdce.move#16

Descriptions:

On Sui, `CoinMetadata` contains crucial information about a token, such as name and symbol. If this information is altered, some users might perceive it as a new token, potentially causing confusion among users.

Suggestion:

It is recommended to freeze `CoinMetadata` if the coin information will not be modified in the future.

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.