Canopy Cornucopia

Audit Report

contact@bitslab.xyz

https://twitter.com/movebit_

Tue Feb 18 2025

Canopy Cornucopia Audit Report

1 Executive Summary

1.1 Project Information

Description	This system facilitates the bridging and deployment of assets from Ethereum vaults into the Cornucopia program on the Move chain. It manages the lockup period and rewards distribution, ensuring gradual unlocking of rewards over time. The system also handles address mapping between Ethereum and Movement, allowing users to claim rewards and withdraw assets after the lockup period.
Type	DeFi
Auditors	MoveBit
Timeline	Mon Feb 03 2025 - Wed Feb 05 2025
Languages	Move
Platform	Movement
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/Canopyxyz/cornucopia-move
Commits	3b5ea2e7c5065ddf9965b3eb30b6f6cd29642be7 904d18e577c910eca5fd5a0cfba90941076469292 c88adc0a9274f2265fd974a47f9d02072ce30b8 db153a401c827d89cf30f3951742fdbf41841d27 aace89e255dcedbf620ef1165b232fe43172366931 ed6e10f917c17378d23eb7db28ec5028088e08

1.2 Files in Scope

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

ID	File	SHA-1 Hash
DDE	packages/deployers/single-asset/sources/concrete/dummy_deployer.move	74b1fc61d072dfd8e879735a2311783bebd894a9
ORE	packages/deployers/single-asset/sources/oapp/shared_oapp/oapp_receive.move	c46641e4e2a7462c0653cdf82aa51d6cafae67e3
OST	packages/deployers/single-asset/sources/oapp/shared_oapp/oapp_store.move	925b5202f3941f6ad726f4c6d4192421fa69a189
OCO	packages/deployers/single-asset/sources/oapp/shared_oapp/oapp_core.move	c82cf967cc89ee7aa129c08bf5012c3a9f0b6309
OCO1	packages/deployers/single-asset/sources/oapp/shared_oapp/oapp_compose.move	2716de4cd81bc48db11da4d3dd85b1c4b8a1aabd
OAP	packages/deployers/single-asset/sources/oapp/oapp.move	68db160903da9bfbddd87352c8ce64315053ad85
EEX	packages/deployers/single-asset/sources/entry_extensions.move	cb98dcc231ae93f6f4d47f68ed7f9310a9e42812
SABD	packages/deployers/single-asset/sources/base/single_asset_base_deployer.move	93b22ebb39895abf9d194b260d51149c333f3357
TIM	packages/std/sources/libs/timelocked.move	4058a543a988fbda2dbe447509957077a34d8467

E71	packages/std/sources/libs/eip712.move	c9be9a807f2ac92efdc6e7a2db50041f21745e9b

1.3 Issue Statistic

Item	Count	Fixed	Acknowledged
Total	10	8	2
Informational	6	5	1
Minor	2	1	1
Medium	1	1	0
Major	1	1	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(Optional)

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 Canopy to identify any potential issues and vulnerabilities in the source code of the Canopy-Cornucopia 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 10 issues of varying severity, listed below.

ID	Title	Severity	Status
DAB-1	Same Error Code	Informational	Fixed
MDE-1	Integer Overflow Risk in `message_decoding`	Medium	Fixed
OAP-1	Lack of Validation for `from`	Major	Fixed
SAB-1	Potential Reward Allocation Without Contribution	Minor	Acknowledged
SAB-2	Unused Constants	Informational	Fixed
SAB-3	Missing Zero-Value Checks	Informational	Acknowledged
SAB-4	Incorrect Comment	Informational	Fixed
SAB-5	Missing Length Check	Informational	Fixed
SDE-1	Exposed Internal Creation Function	Minor	Fixed
TIM-1	Code Optimization	Informational	Fixed

3 Participant Process

Here are the relevant actors with their respective abilities within the Canopy-Cornucopia Smart Contract :

Admin Functions

- Admins can `initialize_deployer` to create new deployer instances.
- Admins can `set_reward_fractions` to configure reward distribution (U12 fixed-point precision).
- Admins can `propose_emergency_withdraw` to initiate emergency withdrawal (timelocked).
- Admins can `execute_emergency_withdraw` to finalize emergency withdrawal after timelock.
- Admins can `update_timelocked_lock_duration` to modify lockup period (timelocked).
- Admins can `transfer_ownership` to transfer deployer ownership.
- Admins can `accept_ownership` to complete ownership transfer.
- Admins can `timelocked_pause` to pause deposit notifications during pre-deployment phase.
- Admins can `sweep_assets` to collect protocol fees to designated address.

User Functions

- Users can `notify_funds` to register bridged assets (OApp-verified).
- Users can `claim_relevant_assets` to withdraw assets (base, shares, rewards).
- Users can `configure_move_claim_address` to set Move-side recipient address.
- Users can `deploy_funds` to invest assets into target protocol (e.g. Liquidswap).
- Users can `batch_deploy_funds` for multiple deployment operations.
- Users can `get_remaining_entitled_amounts` to query claimable balances.
- Users can `get_total_rewards_vested` to check unlocked reward amounts.
- Users can `get_move_claim_addresses` to verify configured addresses.

4 Findings

DAB-1 Same Error Code

Severity: Informational

Status: Fixed

Code Location:

packages/deployers/dual-asset/sources/base/dual_asset_base_deployer.move#117

Descriptions:

`TRACE_DEPLOYER_PAUSED` and `TRACE_MISMATCHED_VECTORS` use the same error code and may not be distinguishable when the contract throws an exception.

Suggestion:

It is recommended to use the correct error code.

Resolution:

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

MDE-1 Integer Overflow Risk in `message_decoding`

Severity: Medium

Status: Fixed

Code Location:

packages/std/sources/libs/message_decoding.move#103

Descriptions:

The current implementation performs a multiplication operation (`contributors_len` * 32) after converting `contributors_len` to u64, which could lead to integer overflow. While `contributors_len` is constrained by `safe_u256_to_u64` to fit in u64, multiplying it by 32 (fixed element size) could exceed u64::MAX (18446744073709551615), causing wrapping behavior and incorrect array bounds calculation.

Suggestion:

It is recommended to implement pre-multiplication range checking to ensure `contributors_len` ≤ (u64::MAX / 32) before performing the multiplication.

Resolution:

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

OAP-1 Lack of Validation for `from`

Severity: Major

Status: Fixed

Code Location:

packages/deployers/dual-asset/sources/oapp/oapp.move#248;

packages/deployers/dual-asset/sources/oapp/oapp.move#248

Descriptions:

The `handle_batch_contribution_message()` method seems to lack verification of the from address, which may lead to problems.

Suggestion:

It is recommended to modify the code like `handle_claim_config_message()` to fix this issue.

Resolution:

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

SAB-1 Potential Reward Allocation Without Contribution

Severity: Minor

Status: Acknowledged

Code Location:

packages/deployers/single-asset/sources/base/single_asset_base_deployer.move#828

Descriptions:

The `set_reward_fractions` function allows setting reward fractions for Ethereum addresses that have no recorded contributions in the contributions table. This creates a potential state where rewards appear to be allocated but cannot actually be claimed, as the claiming process requires contributions to be present. This design might lead to confusion and unintended reward allocations.

Suggestion:

It is recommended to add a check in the `set_reward_fractions` function to verify that the Ethereum address exists in the contributions table before allowing reward fractions to be set, ensuring that rewards are only allocated to addresses that have actually contributed to the protocol.

SAB-2 Unused Constants

Severity: Informational

Status: Fixed

Code Location:

packages/deployers/single-asset/sources/base/single_asset_base_deployer.move#38

Descriptions:

There are multiple unused constants in the contract. Below are some examples,

`E_DEPLOYER_EXISTS` `E_INVALID_DEPOSIT_ASSET` `E_INVALID_PROTOCOL_ID` `E_EXCEEDS_UNNOTIFIED_BALANCE` `E_INVALID_DEPLOYER_BYTES` `E_IN_LOCKUP_PHASE` `E_INVALID_MOVE_ADDRESS_LENGTH`

Suggestion:

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

Resolution:

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

SAB-3 Missing Zero-Value Checks

Severity: Informational

Status: Acknowledged

Code Location:

packages/deployers/single-asset/sources/base/single_asset_base_deployer.move#509

Descriptions:

Some operations do not verify zero or empty states (e.g., no check for non-empty vectors in notify_funds, no existing proposal check in `propose_emergency_withdraw`, and no `total_base_received` zero value validation in `process_claim`), which may lead to unexpected errors or undefined behavior.

Suggestion:

It is recommended to implement explicit checks for zero values and empty vectors to ensure safe

SAB-4 Incorrect Comment

Severity: Informational

Status: Fixed

Code Location:

packages/deployers/single-asset/sources/base/single_asset_base_deployer.move#658

Descriptions:

The code assertion `current_shares_balance > pre_deploy_shares_balance` contradicts its accompanying comment stating "shares balance only increased or stayed the same".

Suggestion:

Updating the comment will improve the clarity and maintainability of the code.

Resolution:

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

SAB-5 Missing Length Check

Severity: Informational

Status: Fixed

Code Location:

packages/deployers/single-asset/sources/base/single_asset_base_deployer.move#729

Descriptions:

In the `set_reward_fractions()`, there is no check if the length of the vector is greater than 0.

Suggestion:

It is recommended to add a check.

Resolution:

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

SDE-1 Exposed Internal Creation Function

Severity: Minor

Status: Fixed

Code Location:

packages/deployers/single-asset/sources/concrete/satay_deployer.move#37

Descriptions:

The `create_internal` function is currently declared as public but serves as an internal helper function only used by the module's entry function `create` and test utilities.

Suggestion:

It is recommended to restrict the visibility of the function.

Resolution:

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

TIM-1 Code Optimization

Severity: Informational

Status: Fixed

Code Location:

packages/std/sources/libs/timelocked.move#132

Descriptions:

In the `manage_timelocked_update()` function, the `cancel_update()` method should be executed regardless of whether the current value is equal to the new value, so there's no need to call cancel_update twice within the conditional statement.

Suggestion:

It is recommended to update the code to fix this issue.

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.