Navi Protocol III

Audit Report

contact@bitslab.xyz

https://twitter.com/movebit_

Thu Feb 06 2025

Navi Protocol III Audit Report

1 Executive Summary

1.1 Project Information

Description	Navi is a liquidity protocol, which supports borrow/lend top assets class in the Move ecosystem
Type	DeFi
Auditors	MoveBit
Timeline	Mon Jan 20 2025 - Thu Feb 06 2025
Languages	Move
Platform	Sui
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/naviprotocol/protocol
Commits	d46b85fc8b3b52901aa9eddbc83660a47a3a044b dd8d847c5587fcc307a2346e3951d2b1a5e206eb

1.2 Files in Scope

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

ID	File	SHA-1 Hash
IV2	lending_core/sources/incentive_v2.move	3d99c05908d6e9152124e7d3e4974a66fb09b4b6
CON1	lending_core/sources/constants.move	f63912999332ad2b140b74055e0800ded13cbd95
IV3	lending_core/sources/incentive_v3.move	d2bcfacab4e5b2f2b8b37098f89491a8e188b6bc
ERR	lending_core/sources/error.move	efc804fe591b4f391d053a8a23454b06367a055f
MAN	lending_core/sources/manage.move	96c2c7a3e7a8b09e8227bccebdf6de9dde082077

1.3 Issue Statistic

Item	Count	Fixed	Acknowledged
Total	3	2	1
Informational	1	1	0
Minor	1	0	1
Medium	1	1	0
Major	0	0	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 Navi to identify any potential issues and vulnerabilities in the source code of the Navi Protocol 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 3 issues of varying severity, listed below.

ID	Title	Severity	Status
IV3-1	Centralization Risk	Medium	Fixed
IV3-2	Unnecessary `mut` Parameter	Minor	Acknowledged
IV3-3	Missing Entry Keyword for `claim_reward_entry`	Informational	Fixed

3 Participant Process

Here are the relevant actors with their respective abilities within the Navi Protocol Smart Contract :

Owner

- The owner can call the `create_reward_fund` function to initialize a reward fund for the system.
- The owner can call the `deposit_reward_fund` function to deposit funds into the reward pool.
- The owner can call the `withdraw_reward_fund` function to withdraw funds from the reward pool.
- The owner can call the `create_incentive_v3` function to create a new incentive object.
- The owner can call the `create_pool` function to initialize a new pool.
- The owner can call the `create_rule` function to define a set of rules for pool.
- The owner can call the `set_enable_by_rule_id` function to enable or disable specific rules based on their IDs.
- The owner can call the `set_max_reward_rate_by_rule_id` function to set the maximum reward rate for a specific rule.
- The owner can call the `set_reward_rate_by_rule_id` function to configure the reward rate for a specific rule.
- The owner can call the `withdraw_borrow_fee` function to withdraw accumulated fees from borrowing activities.
- The owner can call the `deposit_borrow_fee` function to deposit funds into the borrow fee pool.

User

- Users can call the `entry_deposit` function to deposit funds into the system.
- Users can call the `entry_withdraw` function to withdraw their funds from the system.
- Users can call the `entry_borrow` function to borrow funds from the system.
- Users can call the `entry_repay` function to repay borrowed funds.

- Users can call the `entry_liquidation` function to liquidate positions under specified conditions.

4 Findings

IV3-1 Centralization Risk

Severity: Medium

Status: Fixed

Code Location:

lending_core/sources/incentive_v3.move#201

Descriptions:

Centralization risk was identified in the smart contract:

- Administrators can call `withdraw_reward_fund()` to withdraw all the rewards coins.

Suggestion:

It is recommended that measures be taken to reduce the risk of centralization, such as a multi-signature mechanism.

Resolution:

NAVI has separated the permissions for incentive admin and withdraw incentive fund. The latter uses `StorageAdminCap`, which is a capability with the highest security level.

IV3-2 Unnecessary `mut` Parameter

Severity: Minor

Status: Acknowledged

Code Location:

lending_core/sources/incentive_v3.move#676

Descriptions:

The `ger_user_claimable_rewards` function determines how many reward coins the user can claim, however, it does not need the `&mut storage` type parameter, it is recommended to use `&Pool` directly.

Suggestion:

It is recommended to use `&storage` directly.

Resolution:

NAVI comment: This is a historical design where `&mut` was used for getters in the storage module. We have verified that the storage state remains unmodified in the function.

IV3-3 Missing Entry Keyword for `claim_reward_entry`

Severity: Informational

Status: Fixed

Code Location:

lending_core/sources/incentive_v3.move#754

Descriptions:

Although `claim_reward_entry` is named as if it were an entry function, it lacks the actual entry keyword, preventing invocation via SDK or CLI.

Suggestion:

It is recommended to add the `entry` keyword to this function if it is intended to be publicly callable.

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.