Haedal Hmm

Audit Report

MOVEBIT

contact@bitslab.xyz

https://twitter.com/movebit_

Wed Oct 22 2025

Haedal Hmm Audit Report

1 Executive Summary

1.1 Project Information

Description	Haedal is a prime liquid staking protocol natively built on Sui. It provides users with robust liquid staking infrastructure, allowing anyone to stake their SUI & WAL tokens to contribute to the governance and decentralization of the network, while earning continual consensus rewards and unleashing LST liquidity to be used in DeFi On top of its liquid staking protocol, Haedal is also building a series of simple yield products including Haedal Market Maker and more, which generate continuous additional on-chain yields for Haedal and its LST ecosystem Haedal serves as a core pillar of the Sui DeFi by merging native liquid staking and yield strategies with user-friendly accessibility. Aim to empower users to maximize capital efficiency through innovative liquid staking and algorithmic DeFi yield solutions, and build Haedal into the ultimate place to stake and earn on Sui
Type	Staking

Auditors	MoveBit
Timeline	Tue Aug 26 2025 - Wed Oct 22 2025
Languages	Move
Platform	Sui
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/haedallsd/haedal-pmm-contract
Commits	c3aff0425b31c1436de46b3c6f92f8aa915d69f0 4ed725cc6315e903efb16022dadd784ab080b921 1fe6734e85990de065a8f54b7b076f1ef1996a5e

1.2 Files in Scope

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

ID	File	SHA-1 Hash
SCR	sources/script.move	15e621c088470b13a217dfd8d0a920ee047715bb
SMA	sources/safe_math.move	5e828d450ea6ff2414a3a1bade9ad434e892f015
VER	sources/version.move	1bb46fe163350684f73e4dea627723a468346634
ORA	sources/oracle.move	fa3e2e3cda34be627dc2787828069ea7902aa9ff
LPR	sources/liquidity_provider.move	27e9ddf1b6977bbb58e1baaef1528024641acc76
TRA	sources/trader.move	2e4d415e90ed2681635733100012 25997f9e7f71
SET	sources/settlement.move	0ac028612e4f1545b5c466b3ea8436e85d5a6e1b
MOV	Move.toml	4eb1b048c1259fcb1bd6cb079aa28f92a2deb17a
ROB	sources/robot.move	ca85eabc0687a77a35083127e1eb0bcc2d0e29af
ADM	sources/admin.move	20b2d39f585ff20e634c35c7c2fa66b0ea6fdea6

OWN	sources/ownable.move	4d0b8ea864e9ed964c85b889747b2cbab888e783
BRE	sources/breaker.move	5877c08c90fdf3daf669b47d8f13940a960b0adf
ODP	sources/oracle_driven_pool.move	934d68b289ea93504374d0e12bd66069986c1120

1.3 Issue Statistic

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

ID	Title	Severity	Status
ADM-1	Missing Closed-State Check Allows Modification Of Closed pool	Medium	Fixed
OWN-1	Duplicate Error Codes Make Error Source Indistinguishable Description	Informational	Fixed

3 Participant Process

Here are the relevant actors with their respective abilities within the Haedal Hmm Smart Contract :

Admin:

- `set_maintainer` Assign a new maintainer address for a pool.
- `set_liquidity_provider_fee_rate` Set the liquidity provider fee rate.
- `set_liquidity_provider_fee_rate_for_robot` Reserved for robot configuration; currently disabled.
- `set_k` Update the pool's curve parameter `k`.
- `set_k_for_robot` Reserved for robot configuration; currently disabled.
- `set_protocol_fee_rate` Set the protocol's transaction fee rate.
- `set_protocol_fee_rate_for_robot` Reserved for robot configuration; currently disabled.
- `set_quote_usd_price_age` Set the max valid age for quote price data.
- `set_base_usd_price_age` Set the max valid age for base price data.
- `disable_all` Disable all pool operations including deposits and trades.
- `enable_all` Enable all pool operations.
- `disable_trading` Pause all trading activities.
- `enable_trading` Resume trading activities.
- `disable_deposit_quote` Disable quote token deposits.
- `enable_deposit_quote` Enable quote token deposits.
- `disable_deposit_base` Disable base token deposits.
- `enable_deposit_base` Enable base token deposits.
- `disable_buying` Disable buy-side trading.

- `enable_buying` Enable buy-side trading.
- `disable_selling` Disable sell-side trading.
- `enable_selling` Enable sell-side trading.
- `set_base_balance_limit` Set the maximum base token balance limit for the pool.
- `set_quote_balance_limit` Set the maximum quote token balance limit for the pool.
- `share_acl` Create and share an `ACL` object for off-chain access control.
- `add_breaker_to_acl` Add a breaker address to the ACL.
- `del_breaker_to_acl` Remove a breaker address from the ACL.
- `add_robot_to_acl` Add a robot address to the ACL.
- `del_robot_to_acl` Remove a robot address from the ACL.
- `is_breaker` Check whether an address is a breaker.
- `is_robot` Check whether an address is a robot.
- `set_quote_usd_price_age_v2` : Updates the maximum allowed age of the quote token's USD price to control price data freshness.
- `set_base_usd_price_age_v2` : Updates the maximum allowed age of the base token's USD price to prevent trading with stale data.
- `disable_trading_v2` : Disables trading for the specified pool, used for emergency stops or risk control.
- `set_base_balance_limit_v2` : Sets the maximum balance limit for the base token in the pool to prevent asset concentration.
- `set_quote_balance_limit_v2` : Sets the maximum balance limit for the quote token in the pool to control liquidity risk.

4 Findings

ADM-1 Missing Closed-State Check Allows Modification Of Closed pool

Severity: Medium

Status: Fixed

Code Location:

sources/admin.move#285-314

Descriptions:

In the following functions:

- `set_quote_usd_price_age_v2`
- `set_base_usd_price_age_v2`
- `disable_trading_v2`

The contract allows a holder of BreakerCap to modify critical pool parameters (quote/base price age, trading permission). However, these functions only call `pool.assert_version()` without enforcing `pool.assert_not_closed()`.

As a result, even if a pool has already been closed, a caller can still modify its parameters, undermining the design assumption that closed pools should remain immutable.

Suggestion:

Add closed-state validation in each function:

```
pool.assert_not_closed();
```

Resolution:

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

OWN-1 Duplicate Error Codes Make Error Source Indistinguishable Description

Severity: Informational

Status: Fixed

Code Location:

sources/ownable.move#10-11

Descriptions:

In the contract, error codes are defined as:

```
const EPoolExists: u64 = 1;
const EPoolNotExists: u64 = 2;
const EDataNotMatchProgram: u64 = 2;
```

Here, both `EPoolNotExists` and `EDataNotMatchProgram` share the same error code 2. This makes it impossible to distinguish whether an `abort(2)` is triggered because the pool does not exist or due to a program version mismatch.

Suggestion:

Ensure each error code is unique.

```
const EPoolExists: u64 = 1;
const EPoolNotExists: u64 = 2;
const EDataNotMatchProgram: u64 = 3;
```

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.