MOVEBIT
Securing the Move Ecosystem

# MovEx Smart Contract **Audit Report**


https://twitter.com/movebit_

contact@movebit.xyz

# MovEx Smart Contract Audit Report



# 1 Executive Summary

## 1.1 Project Information

| | |
|---|---|
| **Description** | An AMM DEX on Sui. |
| **Type** | DEX |
| **Auditors** | MoveBit |
| **Timeline** | Apr 26, 2023 – May 10, 2023 |
| **Languages** | Move |
| **Platform** | Sui |
| **Methods** | Architecture Review, Unit Testing, Manual Review |
| **Source Code** | https://github.com/MoveExchange/MovEX-swap-v2 |
| **Commits** | bee12a9e215f242e8501ef1f8123fe82c7dac031<br>d033babe6b92202ddf400eedc6d9327831cedf1b |

## 1.2 Files in Scope

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

| ID | Files | SHA-1 Hash |
|---|---|---|
| POL | sources/pool.move | dfc0a72e2d91cf64feb70410bac60ebebef7cdff |

## 1.3 Issue Statistic

| Item | Count | Fixed | Acknowledged |
| --- | --- | --- | --- |
| Total | 17 | 14 | 3 |
| Informational | 2 | 2 | |
| Minor | 6 | 3 | 3 |
| Medium | 5 | 5 | |
| Major | 4 | 4 | |
| Critical | | | |

## 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 sees 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 **MovEx** to identify any potential issues and vulnerabilities in the source code of the **MovEx** 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 have identified **17** issues of varying severity, listed below.

| ID | Title | Severity | Status |
| --- | --- | --- | --- |
| POL-01 | Can Create Pools With the Same Type | Medium | Fixed |
| POL-02 | Lack of Vector Coin Support | Minor | Acknowledged |
| POL-03 | Lack of K-Value Check | Medium | Fixed |
| POL-04 | Inconsistent Token Ratios in Adding Liquidity | Major | Fixed |
| POL-05 | LSP Value Should be Greater than 0 | Major | Fixed |
| POL-06 | Incorrect Comment on `Fee_percent` | Informational | Fixed |
| POL-07 | Create pools with the same coin type | Medium | Fixed |
| POL-08 | Lack of Minimum Liquidity Requirement | Medium | Fixed |
| POL-09 | No limit to swap | Major | Fixed |
| POL-10 | Missing Emit Events | Minor | Fixed |
| POL-11 | Incorrect Protocol Fee Handling | Medium | Fixed |
| POL-12 | Unnecessary `store` Ability | Minor | Acknowledged |
| POL-13 | `change_order` Function Design Flaw | Minor | Fixed |
| POL-14 | `fee_percent` Redundant Judgment Condition | Informational | Fixed |
| POL-15 | Improve Precision With `sqrt_u128` | Minor | Acknowledged |
| POL-16 | No Minimum Liquidity Locked | Major | Fixed |
| POL-17 | K Value Check Condition Error | Minor | Fixed |

# 3 Participant Process

Here are the relevant actors with their respective abilities within the `MovEx` SmartContract：

**Admin**

- Admin can `withdraw_protocol_fee` .
- Admin can `reset_fee_percent` .

**User**

- User can `create_pool` .
- User can `swap_token1&2` .
- User can `add_liquidity_` .
- User can `remove_liquidity_` .
- User can `get_input_price` .

# 4 Findings

## POL-01 Can Create Pools With the Same Type

**Severity: Medium**

**Status: Fixed**

**Code Location:** sources/pool.move#L156

**Descriptions:** The `create_pool` function does not judge whether the same type of pool already exists before, which will lead to the existence of the same pool at the same time and the depth of each pool is not large enough, causing the problem of transaction slippage.

**Suggestion:** Determine whether a pool of the same type exists before creating a pool.

**Resolution:** The project team updated the codes in commit `39f4e635087786a4caa96e47c5d3b7f0b4745d3e` and resolved this issue.

## POL-02 Lack of Vector Coin Support

**Severity: Minor**

**Status**: **Acknowledged**

**Code Location:** sources/pool.move#L244.

**Descriptions:** Because of the object model of sui, the entry function can support the coin in the vector, thus supporting the transaction of multiple coins.

**Suggestion:** Use the `pay` module under sui to realize the merge operation of multiple vector coins.

## POL-03 Lack of K-Value Check

**Severity: Medium**

**Status**: **Fixed**

**Code Location:** sources/pool.move#L275

**Descriptions:** According to the Constant Product Market Maker Model of AMM, the product of the quantity of the two tokens in the pool will increase each time the swap is completed, and the excess part is the handling fee for the liquidity provider. So executing swap, contract should verify that the K value is greater than the previous K value. It is recommended to add assert verification.

**Suggestion:** Add the assert to limit the value of k to incremental.

**Resolution:** The project team updated the codes in commit `a5f3de2ab5521899fb8fac2f2f613c36863e5788` and resolved this issue.

## POL-04 Inconsistent Token Ratios in Adding Liquidity

**Severity: Major**

**Status**: **Fixed**

**Code Location:** sources/pool.move#L317

**Descriptions:** When adding liquidity, the number of liquidity tokens that users can obtain is calculated based on the ratio of the added tokens to the pool, so adding liquidity, the excess money should be returned to the user, and should not be added to the pool.

**Suggestion:** Return the excess coins to the user.

**Resolution:** The project team updated the codes in commit `81f05ef597342b259e9db69f510c90d66f07cbe6` and resolved this issue.

## POL-05 LSP Value Should be Greater than 0

**Severity: Major**

**Status: Fixed**

**Code Location:** sources/pool.move#L323

**Descriptions:** When adding liquidity, it is necessary to ensure that the number of liquidity tokens returned at the end is greater than 0, otherwise, the user can not get the tokens back.

**Suggestion:** Increase the assert to limit the number of `LSPs` to be greater than 0.

**Resolution:** The project team updated the codes in commit `a5f3de2ab5521899fb8fac2f2f613c36863e5788` and resolved this issue.

## POL-06 Incorrect Comment on `fee_percent`

**Severity: Informational**

**Status: Fixed**

**Code Location:** sources/pool.move#L125

**Descriptions:** `fee_percent` is set to 0–10000, and 1000 corresponds to 10%, while 1 corresponds to 0.01%, but the corresponding value in the comment is incorrect.

**Suggestion:** Modify to the correct value, 10% and 0.01%.

**Resolution:** The project team updated the codes in commit `a5f3de2ab5521899fb8fac2f2f613c36863e5788` and resolved this issue.

## POL-07 Can Create Pools With the Same Coin Type

**Severity: Medium**

**Status: Fixed**

**Code Location:** sources/pool.move#L156

**Descriptions:** For a pool, if the two types of tokens are the same, the pool can be considered meaningless.

**Suggestion:** When creating a pool, limit the two types of tokens to be different.

**Resolution:** The project team updated the codes in commit `03c5303d5613d7a12fa5a9c53a98ac1aa8451e22` and resolved this issue.

## POL-08 Lack of Minimum Liquidity Requirement

**Severity: Medium**

**Status: Fixed**

**Code Location:** sources/pool.move#L156

**Descriptions:** While initially minted shares are equal to the geometric mean of the amount deposited, the value of liquidity pool shares may grow over time, either by accumulating transaction fees or by contributing to the liquidity pool. In theory, this could lead to the smallest number of liquidity pool shares (1e-9 pool shares) being so valuable that small liquidity providers cannot provide any liquidity.

**Suggestion:** A minimum liquidity requirement should be implemented during the first liquidity provision to prevent malicious users from exploiting the system. This will require users to deposit a minimum amount of liquidity, ensuring that the liquidity pool is balanced and operating smoothly.

**Resolution:** The project team updated the codes in commit `a5f3de2ab5521899fb8fac2f2f613c36863e5788` and resolved this issue.

## POL-09 No Limit to Swap

**Severity: Major**

**Status: Fixed**

**Code Location:** sources/pool.move#L201

**Descriptions:** Due to the delay of the blockchain transaction, the price when the user sends the transaction and the actual execution of the contract may be different, which may cause the amount of exchange to be lower than expected, resulting in the loss of user funds.

**Suggestion:** When exchanging, limit a minimum `min_out` to limit the result of the exchange to be greater than or equal to this `min_out`.

**Resolution:** The project team updated the codes in commit `a5f3de2ab5521899fb8fac2f2f613c36863e5788` and resolved this issue.

## POL-10 Missing Emit Events

**Severity: Minor**

**Status: Fixed**

**Code Location:** sources/pool.move#L109

**Descriptions:** It is recommended to throw an update event when `fee_percent` is updated.

**Suggestion:** It is recommended to throw an update event of fee_percent.

**Resolution:** The project team updated the codes in commit `a5f3de2ab5521899fb8fac2f2f613c36863e5788` and resolved this issue.

## POL-11 Incorrect Protocol Fee Handling

**Severity: Major**

**Status: Fixed**

**Code Location:** sources/pool.move#L84

**Descriptions:** According to the logic, the protocol fee is a percentage of the total fee, but if it is not withdrawn, it will be withdrawn by the person who adds liquidity, resulting in a protocol fee of 0. This part of the fee extraction may also cause asset losses for some users who provide liquidity.

**Suggestion:** Modify the code logic to handle this part of the fee properly, such as directly withdrawing it to the fee recipient or storing it in some other address.

**Resolution:** The project team updated the codes in commit `a5f3de2ab5521899fb8fac2f2f613c36863e5788` and resolved this issue.

## POL-12 Unnecessary `store` Ability

**Severity: Minor**

**Status: Acknowledged**

**Code Location:** sources/pool.move#L77

**Descriptions:** The event structure in Sui needs to have the ability to `copy` and `drop`, and does not need the `store` ability.

**Suggestion:** Delete the attribute of the structure `store`.

## POL-13 `change_order` Function Design Flaw

**Severity: Minor**

**Status: Fixed**

**Code Location:** sources/pool.move#L533

**Descriptions:** In the `change_order` function, if the `TypeName` of the two tokens to be processed is the same, an empty array will be `pop_backed` and an error will be reported. At the same time, an error will be reported when the `TypeName` of the two tokens is the same but the length is not equal.

**Suggestion:** Make sure the two token types are different.

**Resolution:** The project team updated the codes in commit `d033babe6b92202ddf400eedc6d9327831cedf1b` and resolved this issue.

## POL-14 `fee_percent` Redundant Judgment Condition

**Severity: Informational**

**Status: Fixed**

**Code Location:** sources/pool.move#L236,L158,L159

**Descriptions:** In `assert!(fee_percent >= 0 && fee_percent < 10000, EWrongFee);` code `fee_percent` is u64, so the range is always `>=0`, so the first condition is not necessary.

**Suggestion:** Remove redundant conditions.

**Resolution:** The project team updated the codes in commit `d033babe6b92202ddf400eedc6d9327831cedf1b` and resolved this issue.

## POL-15 Improve Precision With `sqrt_u128`

**Severity: Minor**

**Status: Acknowledged**

**Code Location:** sources/pool.move#L236,L158,L159

**Descriptions:** In `Sui Framework`, there are two kinds of inclusions, `sqrt`, and `sqrt_u128`. When calculating share, multiplying two `u64` can use `sqrt_u128` to improve the accuracy.

**Suggestion:** Improve precision with `sqrt_u128`.

## POL-16 No Minimum Liquidity Locked

**Severity: Major**

**Status: Fixed**

**Code Location:** sources/pool.move#L236,L158,L159

**Descriptions:** When creating a pool to add liquidity, in addition to limiting a minimum liquidity value, this minimum liquidity share also needs to be locked, because when the last user removes liquidity, lp_supply is updated to 0, resulting in abnormalities in `swap` and `add_liquidity`.

**Suggestion:** Transfer minimal liquidity to a locked address.

**Resolution:** The project team updated the codes in commit `89e31cc44e25fe4f57190b6e85571ed7919664dc` and resolved this issue.

## POL-17 K Value Check Condition Error

**Severity: Minor**

**Status: Fixed**

**Code Location:** sources/pool.move#L321

**Descriptions:** At the end of the swap function, the value of k in the pool will be limited to an increase, but this is for the case of handling fees. When `fee_percent` is set to 0, `swap` will not change the value of k in the pool, so when the limit k is increased, the handling fee Pools with a value of 0 cannot be swapped normally, and the condition of `liquidity_before_swap < liquidity_after_swap` should be changed to `<=`.

**Suggestion:** Modify to the correct condition.

**Resolution:** The project team updated the codes in commit `b5abdb8022b059a37ccce8240f88804c504fc1af` and resolved this issue.

# 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, more information can be found in the *Resolution* section.
- **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.

MOVEBIT

Securing the Move Ecosystem

https://twitter.com/movebit_

contact@movebit.xyz