FlowX

Audit Report

MOVEBIT

✉ contact@bitslab.xyz

https://twitter.com/movebit_

Wed May 08 2024

FlowX Audit Report

1 Executive Summary

1.1 Project Information

Description	FlowX is the ecosystem-focused decentralized exchange built on the Sui Blockchain.
Type	Dex
Auditors	MoveBit
Timeline	Mon Apr 15 2024 - Wed May 08 2024
Languages	Move
Platform	Sui
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/FlowX-Finance/clmm-contracts
Commits	6e34506aba4f2c4bbc3c896b9c9f671e8cf2e00d 040cfd34b4b91687754563b63c8be3b9386f35a6

1.2 Files in Scope

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

ID	File	SHA-1 Hash
ACA	sources/admin_cap.move	9db0d29b39bcffe787434375decf8158d257d139
PMA	sources/position_manager.move	1449d3c31d6ea438bed9312caa6d6269c3560cb2
POS	sources/position.move	6e8626ba34e93c1eabc319d1f07acbb348a7093d
VER	sources/versioned.move	004e34e5c0e58498de66203c22393d1f509668c3
SMA	sources/libs/swap_math.move	788bed12f6f2eb2ea48dca50df173424b03d1060
MU2	sources/libs/math_u256.move	8f26a9ba3561514556d2472df7f0247230575097
TUT	sources/libs/test_utils.move	efbcc1b750924efdbac5b865f56420c0ed626547
FMU6	sources/libs/full_math_u64.move	84b7b662cd4c518db038242c89165e9277e0ae54
FMU1	sources/libs/full_math_u128.move	c46c0eea83a82d5a4686ebe1406b61bf336fb264
LMA	sources/libs/liquidity_math.move	c9a6708ee76a00f27ee9e9a439c2ce02218e577b
CON	sources/libs/constants.move	6d84c684f7ef552344eb4810e5856a72752150b4

COM	sources/libs/comparator.move	66fdcd1fec7cac5ad1ca5348f11c603c454642c6
CAS	sources/libs/caster.move	4f0cfe539fa48915b24a078aa08b1400ed4bbaf8
TBI	sources/libs/tick_bitmap.move	84301a8689ccb28373c9b9a4209cfa76a52106c5
SPM	sources/libs/sqrt_price_math.move	bf06a4033109615853f974f11be34bad1e5a0202
I32	sources/libs/i32.move	cbd21adb2c658722e64d10d2342cdb1f0fa7f843
UTI	sources/libs/utils.move	b2d515c2061d5df7b7611c9f24567cfeec7067e1
I12	sources/libs/i128.move	7dc6a3933e2829feb651fd7dee63fde8be8fd152
I64	sources/libs/i64.move	936a6c2fabdd01585512c0ad548d079d3bec1564
BMA	sources/libs/bit_math.move	c2ed88b7c5aba8396e352efbe2989bdcfee11c27
TMA	sources/libs/tick_math.move	663431610073d3a4ca20a9b0f890d92305b2f11a
POO	sources/pool.move	b9aff6db8f7d56736f90e01f16d880fe02728b88
SRO	sources/swap_router.move	da0c82dae13d8012837226b3aae67b4eb38fa739
ORA	sources/oracle.move	441a3835e48985d9c0419b13ec9cdede274ddc38

TIC	sources/tick.move	f484489391714c9d7e96b27570a6eb94cbfdcbfa
PMA1	sources/pool_manager.move	eb0e317b43df4ff7ceef2729a8323bfcfc75e0b5
PMA	sources/position_manager.move	50f20547bd361ca294e672c5ab861da9060163db
POS	sources/position.move	8fee7a69a210e0e9c642b987d69ae47dfdc265e0
SMA	sources/libs/swap_math.move	5b5c744f3f14e6f9cc87a2d0b13a826792b8033c
CAS	sources/libs/caster.move	97902e1a3c50e2831b74f7bffbc0216ab90f4df5
TBI	sources/libs/tick_bitmap.move	5a79ed0ac3b417b19ffe34d2b814d956b1e9f969
BMA	sources/libs/bit_math.move	c865ea52408e9274b1d4ad16948d9532e752e969
POO	sources/pool.move	4ab6118f6c075224d49156edc47dc338538647b6
ORA	sources/oracle.move	39fa8692a59f1e7cd4ad231918b6355eb7518817
TIC	sources/tick.move	880810f81c74b31df24350cf2595e499035828a1

1.3 Issue Statistic

Item	Count	Fixed	Acknowledged
Total	6	6	0
Informational	2	2	0
Minor	2	2	0
Medium	0	0	0
Major	2	2	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

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 FlowX to identify any potential issues and vulnerabilities in the source code of the clmm-contracts 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 6 issues of varying severity, listed below.

ID	Title	Severity	Status
PMA-1	`collect` Parameter Checking May Fail	Minor	Fixed
POO-1	`initialize` Missing Version Checks	Major	Fixed
POO-2	May Be Wrong Parameters In `flash`	Major	Fixed
POO-3	Can Pass Token When Burning Liquidity	Informational	Fixed
VER-1	Lack of Events Emit	Minor	Fixed
VER-2	Unnecessary `check_version`	Informational	Fixed

3 Participant Process

Here are the relevant actors with their respective abilities within the clmm-contracts Smart Contract :

Owner

- The owner can collect the protocol fee accrued to the pool by the `collect_protocol_fee` function.
- The owner can set the protocol fee rate by the `set_protocol_fee_rate` function.
- The owner can set the initial reward for the pool by the `initialize_pool_reward` function.
- The owner can increase the amount of rewards allocated to the pool by the `increase_pool_reward` function.
- The owner can extend reward allocation time by the `extend_pool_reward_timestamp` function.
- The owner can can add new fee_rate and tick_spacing by the `enable_fee_rate` function.

User

- Users can open or close a position by the `open_position` and `close_position` functions.
- Users can swap coin_x for coin_y, or coin_y for coin_x by the `swap` function.
- Users can flash loan coin_x and coin_y from the pool by the `flash` function.
- Users can collect coins owed to a position by the `collect` function.
- Users can collect rewards owed to a position by the `collect_pool_reward` function.
- Users can add or reduce the liquidity of the positions they own by the `increase_liquidity` and `decrease_liquidity` functions.
- Users can create new pools by the `create_pool` or `create_and_initialize_pool` functions.
- Users can set the initial price for the pool by the `initialize` function.

4 Findings

PMA-1 `collect` Parameter Checking May Fail

Severity: Minor

Status: Fixed

Code Location:

sources/position_manager.move#219

Descriptions:

The `collect` function only allows the `count_x_requested` and `count_y_requested` to both be non-zero in order to pass the `check_zero_amount` function and if the user's position has only one token in it, he must be forced to pass one of them as an arbitrary value in order to call the `collect` function.

Suggestion:

It is recommended to use the `or` logic instead of the `and` logic.

Resolution:

The client has already solved the problem based on our recommendations.

POO-1 `initialize` Missing Version Checks

Severity: Major

Status: Fixed

Code Location:

sources/pool.move#341

Descriptions:

There is no version checking when calling `initialize` function, so when the object is upgraded or the code of those functions is modified, the user can still call the old initialize function to initialize the pool which may lead to unexpected results.

Suggestion:

It is recommended to include the Versioned object in the parameter list of any function that can modify the state.

Resolution:

The client has added version checking based on our recommendations.

POO-2 May Be Wrong Parameters In `flash`

Severity: Major

Status: Fixed

Code Location:

sources/pool.move#755-803

Descriptions:

In the `flash` function, if the money borrowed by the user is greater than the existing number of the pool, it will automatically borrow the existing number of the pool of funds, which will lead to the user receiving money not equal to the borrowed money, and the handling fee is calculated by the user's input parameters, and in the `FlashReceipt` given to the user is also the use of the input parameters rather than the value of the actual out of the user, so, the user to repay the money may have been failed and do not know the reason, if the user to return the money succeeded, which may be a huge amount of money to the user's losses.

Suggestion:

It is recommended that users not be allowed to lend more than the available funds in the pool.

Resolution:

The client has changed to the correct code as suggested.

POO-3 Can Pass Token When Burning Liquidity

Severity: Informational

Status: Fixed

Code Location:

sources/pool.move#405

Descriptions:

The `modify_liquidity` function is used to modify the liquidity of a given pool, a sufficient number of x and y tokens need to be passed in to add liquidity, and no tokens should be passed in to remove liquidity, i.e., the number of tokens should be 0. However, when the user removes the liquidity, he can still pass tokens to the pool.

Suggestion:

It is recommended to confirm that this is compatible with the design concept.

Resolution:

The client has already solved the problem based on our recommendations.

VER-1 Lack of Events Emit

Severity: Minor

Status: Fixed

Code Location:

sources/versioned.move#41;

sources/position_manager.move#104;

sources/position_manager.move#211

Descriptions:

The contract lacks appropriate events for monitoring sensitive operations, which could make it difficult to track sensitive actions or detect potential issues.

Suggestion:

It is recommended to emit events for those important functions.

Resolution:

The client has added events based on our suggestions.

VER-2 Unnecessary `check_version`

Severity: Informational

Status: Fixed

Code Location:

sources/versioned.move#34

Descriptions:

In the `check_version_and_upgrade` function, if the `Versioned` object is less than `VERSION`, then `self.version` will be set to the latest `VERSION`, and `check_version` will never throw an exception after that, and if the `Versioned` object is equal to `VERSION`, then `check_version` will not throw an exception either, so `check_version` plays no role in this function.

Suggestion:

It is recommended to optimize the logic of the code.

Resolution:

The client has already solved the problem based on our recommendations.

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.

MOVEBIT VERIFIED
MOVEBIT