YuzuDex Movefun

Audit Report

MOVEBIT

contact@bitslab.xyz

https://twitter.com/movebit_

Fri Feb 14 2025

YuzuDex Movefun Audit Report

1 Executive Summary

1.1 Project Information

Description	A Clmm Dex.
Type	DeFi
Auditors	MoveBit
Timeline	Tue Dec 31 2024 - Fri Jan 17 2025
Languages	Move
Platform	Movement
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/YuzuDEX/CLMM-Core https://github.com/YuzuDEX/MoveFun https://github.com/YuzuFinance/move-fun
Commits	https://github.com/YuzuDEX/CLMM-Core: c653809a05ce053941fc1af6a9e2a9fefad7dc32 c1818ce6527d3428c4818e1376d782c27d9b19f1 a0fea27c5607aeb4205535b7bd4a6c6348f3090a https://github.com/YuzuDEX/MoveFun: cf76eefc825170f805f458791682f2fbeaea82f6 https://github.com/YuzuFinance/move-fun: b7e3b833084c5c80808d02143704883237057 5f5 8a3d1c6c37bb1568a2743b9bff4ed88c3bf76896 fa29f4c0cf518b9b5a655a8c73beaf3cc99ad0f9

1.2 Files in Scope

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

ID	File	SHA-1 Hash
PNM	sources/position_nft_manager.move	45c6de2baf8ce73b6a9aea53b5beba562c0a27bb
ROU	sources/router.move	df8f6e323157187d4513272f41ce66794125fd10
FPO	sources/libs/fixed_point.move	8eb5d529e60daa3e2aebfa8c6c72d25d1bac6c2c
SMA	sources/libs/swap_math.move	3797e86ff375ee4c3125d16809f9094f58e95e6d
UMA	sources/libs/unsafe_math.move	e7ec31c78947697d9dce9dbcc2bec7ad9473e8c0
MAT	sources/libs/math.move	dbfd328a10b86003c1b2d35708b12025ea0bedde
LMA	sources/libs/liquidity_math.move	f55677ba1a5ad56923d394e374e9e61710cde7c6
FHE	sources/libs/fa_helper.move	bbf01a41c32457b59bc5b07cfb59360710076029
CHE	sources/libs/coin_helper.move	09309bfa246100bf71e658fde89ab2b2de16daa6
TBI	sources/libs/tick_bitmap.move	e1afb6a6e68654fec9e638407ae941e181386fe2
SPM	sources/libs/sqrt_price_math.move	fbdc4321c3d934483845f67749e9a4e67ea41c4e

I12	sources/libs/i128.move	14c36cac882fa9060bb229ad94120ff75924f99c
TIC	sources/libs/tick.move	472ca9574e770646a805409e6683e79d8f8b85a5
BMA	sources/libs/bit_math.move	6f65716b15c19fed1e24f7970905c06f36e02451
TMA	sources/libs/tick_math.move	6a8202511bdfd6c37b47be68980040b94df71695
EME	sources/emergency.move	7e56e5a6a4d5aea771beecec02d8d49926485c9e
SCR	sources/scripts.move	c8034bc8f645f84ab7b114ac44f6135d19cf21fc
LPO	sources/liquidity_pool.move	f02a36cb8bbad8feae44696d46060f030098dfc4
CON	sources/config.move	05df1675cdcec73e76ec31c6686966515cbcf41f
FTI	sources/fee_tier.move	1d5be178e27499b3cd40b0c90b574af287aa7cb6
RMA	sources/reward_manager.move	43c3c3e9897c7f46474f190fcbda3243d2aebb75
BCU	core/sources/bonding_curve.move	2e14429a1050eab4413148cda31efb6c730e33ed
ADM	core/sources/admin.move	9dbc52292db0a9975ad784dfa557c6747849e316
MAT	core/sources/math.move	e361f35598e3f5194603dbd59837bb69ae1e5f13

FAL	core/sources/fungible-asset/fungible_asset_launch.move	6b9dd740fc2931b91c171e6a77653587f1099fc6
FAR	core/sources/fungible-asset/fungible_asset_router.move	b13ab510b0e8c3690da2dc8b321d1bdd3ebfc010
FAS	core/sources/fungible-asset/fungible_asset.move	50a894429d0c24c57fe3501ea60bdb6b5facbc27
LCR	core/sources/coin/legacy_coin_router.move	54c97a2024f8f6b55811379d246db545fb7db9e6
LCL	core/sources/coin/legacy_coin_launch.move	9620327e772eb940246ab9b8808bb7d48b9f7634
LCO	core/sources/coin/legacy_coin.move	ded6b947784caf00c01ed325360dd47719626c14
COM	core/sources/common.move	8370f1b499d5f9ddd64ef29f30c231bcf7b38ae5

1.3 Issue Statistic

Item	Count	Fixed	Acknowledged
Total	6	3	3
Informational	2	0	2
Minor	2	1	1
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(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 YuzuDex to identify any potential issues and vulnerabilities in the source code of the YuzuDex Movefun 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
ADM-1	Uninitialized Pending Ownership Transfer Structure	Major	Fixed
ADM-2	Unbounded Fee Setting in `set_buy_fee()` and `set_sell_fee()` Function	Minor	Acknowledged
BCU-1	Strict Equality Condition for Bonding Curve Goal May Prevent Launch	Major	Fixed
BCU-2	Rounding Direction in Fixed-Point Calculation	Minor	Fixed
BCU-3	Unused Function `set_market_cap_goal()` Should Be Marked for Testing Only	Informational	Acknowledged
FAS-1	Potential Decimal Mismatch in Bonding Curve Price Calculation	Informational	Acknowledged

3 Participant Process

Here are the relevant actors with their respective abilities within the YuzuDex Movefun Smart Contract :

Participant Process

Admin

- The Admin can offer admin privileges to a new address through `offer_admin_previliges()` .
- The Admin can offer treasury privileges to a new address through `offer_treasury_previliges()` .
- The Admin can cancel the admin privileges transfer request through `cancel_admin_previliges()` .
- The Admin can cancel the treasury privileges transfer request through `cancel_treasury_previliges()` .
- The Admin can set the creation fee through `set_creation_fee()` .
- The Admin can set the graduation fee through `set_graduation_fee()` .
- The Admin can set the buy fee through `set_buy_fee()` .
- The Admin can set the sell fee through `set_sell_fee()` .

User

- The User can claim admin privileges through `claim_admin_previliges()` . The admin privileges offer is accepted by the receiver.
- The User can claim treasury privileges through `claim_treasury_previliges()` . The treasury privileges offer is accepted by the receiver.
- The User can reject admin privileges through `reject_admin_previliges()` . The offer must be made to the user.
- The User can reject treasury privileges through `reject_treasury_previliges()` . The offer must be made to the user.

4 Findings

ADM-1 Uninitialized Pending Ownership Transfer Structure

Severity: Major

Status: Fixed

Code Location:

core/sources/admin.move#78

Descriptions:

The `Pending<T>` struct is defined to store pending ownership transfers but lacks proper initialization and transfer logic, making it effectively unusable. Without a mechanism to initialize and assign ownership transfer requests, the structure does not serve its intended purpose.

Suggestion:

It is recommended to initialize the `Pending<T>` structure properly and transfer it to an address.

Resolution:

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

ADM-2 Unbounded Fee Setting in `set_buy_fee()` and `set_sell_fee()` Function

Severity: Minor

Status: Acknowledged

Code Location:

core/sources/admin.move#321,312

Descriptions:

The `set_buy_fee()` and `set_sell_fee()` function allows an admin to set the buy fee using `new_numerator` and `new_denominator`. However, there is no validation to ensure the fee remains within a reasonable range (e.g., between 0 and 1). This lack of constraint could lead to unintended or excessive fees, potentially harming users.

Suggestion:

It is recommended to enforce a reasonable range for the buy fee by ensuring `0 ≤ new_numerator ≤ new_denominator`.

BCU-1 Strict Equality Condition for Bonding Curve Goal May Prevent Launch

Severity: Major

Status: Fixed

Code Location:

core/sources/bonding_curve.move#198;

core/sources/fungible-asset/fungible_asset.move#685

Descriptions:

In the `is_bonding_curve_goal_reached()` function, the condition `info.bonding_curve_goal == available_supply` requires that the exact amount of tokens be sold before launch. However, in practice, reaching this precise value can be difficult due to the unpredictable nature of the last order. This could prevent the contract from ever reaching the launch condition, effectively locking the process.

Suggestion:

It is recommended to use `<=` instead of `==` to allow the launch condition to be met when the required threshold is reached or exceeded.

Resolution:

This problem may cause the token to be unable to be listed normally, because the number of aptos obtained from the sale must be equal to a certain value, and this condition may not be met.

BCU-2 Rounding Direction in Fixed-Point Calculation

Severity: Minor

Status: Fixed

Code Location:

core/sources/bonding_curve.move#200

Descriptions:

The function `fixedpoint64::decode_round_up(new_supply_fq64)` is currently rounding up, but in this context, rounding down would be more reasonable. This calculation determines the amount the user should pay:

```
(current_supply as u64 new_supply
```

If `new_supply` is rounded up, the user pays slightly less than they should. If `new_supply` is rounded down, the user pays slightly more, which ensures the protocol is not losing precision, as the user should bear the rounding loss.

Suggestion:

It is recommended to use the rounding down method.

Resolution:

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

BCU-3 Unused Function `set_market_cap_goal()` Should Be Marked for Testing Only

Severity: Informational

Status: Acknowledged

Code Location:

core/sources/bonding_curve.move#163

Descriptions:

The `set_market_cap_goal()` function is not referenced or used anywhere in production code except within test cases. Leaving it accessible in non-test environments may introduce unnecessary attack surfaces or unintended modifications to the bonding curve parameters.

Suggestion:

It is recommended to mark the function with `#[test]` to ensure it is only callable during testing and does not impact production logic.

FAS-1 Potential Decimal Mismatch in Bonding Curve Price Calculation

Severity: Informational

Status: Acknowledged

Code Location:

core/sources/fungible-asset/fungible_asset.move#306,370

Descriptions:

The function `bonding_curve::calculate_price` currently uses `decimals as u128`, which represents `X`'s (meme token's) decimals. However, since the constant `k` in the bonding curve is likely based on `Y`'s (Aptos) decimals, this could introduce inaccuracies if `X` and `Y` have different decimal places. In the current implementation, both `X` and `Y` use `10^8` decimals, making the calculation valid. However, if `X` had a different decimal precision, the price calculation might be incorrect.

Suggestion:

It is recommended to ensure that the decimal value used corresponds to `Y` (Aptos).

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.

MOVEBIT VERIFIED
MOVEBIT