Suiswap
Smart Contract
Audit Report

MOVEBIT

contact@movebit.xyz

https://twitter.com/movebit_

06/09/2023

Suiswap Smart Contract Audit Report

1 Executive Summary

1.1 Project Information

Description	Suiswap is a decentralized token trading platform and exchange built on the SUI blockchain by Vivid Network. It aims to provide a secure, fast, and agile trading environment for the SUI ecosystem.
Type	DEX
Auditors	MoveBit
Timeline	May 24, 2023 – June 6, 2023
Languages	Move
Platform	Sui
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/vividnetwork/suiswap-audit
Commits	a46d60ac35f6a3a08e01be579b3cc6df840de62e 3d1dc12482231b34726668a179123edd2bceb990 3c8da82745fdda1e05cbf127e3368f8b319b3fc2 66795c96f17d87a15c8e6f9f9e546932c1a18d4f

1.2 Files in Scope

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

ID	Files	SHA-1 Hash

PMS	sources/permission.move	531ccb4a588c2138c68b4266e73d705e69d263fa
POL	sources/pool.move	52d6bbe121bac744b78dc6a887053d355b84ed2c
RTO	sources/ratio.move	dcf2a43ae58ba9eff5fd7ce5cbbb25094f5eaee9
SBC	sources/sbalance.move	10f64b4a8762c1c8d1753c0f657b2c486566ef3d
TKN	sources/token.move	aaacbbfee72cc2d12e94af0d3d196a0b12aee3a2
UTL	sources/utils.move	6d177eac94f38f6d8ee1310f860117ec534f68f4
VPT	sources/vpt.move	fdfdc8cbe9a7b70e83a132a755e7aea130ce16d7

1.3 Issue Statistic

Item	Count	Fixed	Partially Fixed	Acknowledged
Total	20	17	1	2
Informational	2	2		
Minor	9	7		2
Medium	4	3	1	
Major	5	5		
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 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 **Vivid Network** to identify any potential issues and vulnerabilities in the source code of the **Suiswap** 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 **20** issues of varying severity, listed below.

ID	Title	Severity	Status
VPT-01	Unused Constant	Minor	Fixed
VPT-02	Unnecessary Comparison of Boolean Values	Minor	Fixed
TKN-01	Lack of Event Logging in `do_withdraw_token_bank_admin_balance` Function	Minor	Fixed
TKN-02	Lack of Permission Verification in `do_add_token_ido_whitelist` Function	Major	Fixed
TKN-03	Potential Event Bypass in `do_claim_token_airdrop_token_legacy` Function	Minor	Fixed

TKN-04	Centralization Risk in `do_increase_token_supply` Function	Medium	Partially Fixed
TKN-05	Missing Separation of Production and Test Code for `token.move`	Major	Fixed
TKN-06	Add Assertion Validation for `share_minted`	Minor	Fixed
TKN-07	Assertion Error in `do_swap_x_to_y_direct` and `do_swap_x_to_y_direct` Functions	Medium	Fixed
TKN-08	Duplicate Code in `do_claim_token_airdrop_token_legacy` Function	Informational	Fixed
TKN-09	Incorrect `token_type` Values in Recorded Events in `do_send_staked_token` Function	Minor	Fixed
TKN-10	Lack of Validation for `do_create_registry` and `do_create_token_farm` Functions	Minor	Fixed
POL-01	Missing Emit Event	Minor	Acknowledged
POL-02	Missing Total Liquidity Restriction and Minimum Locking for `add_liquidity_direct_impl` Function	Major	Fixed
POL-03	The initialization value of `admin_fee` in the `do_create_registry` function is inconsistent with the comment	Informational	Fixed

POL-04	Incorrect Calculation of `th_fee` Due to Deducting `admin_fee` from `balance` Before Computing `th_fee`	Minor	Acknowledged
POL-05	Incorrect Loop Termination and Inconsistent Implementation in `ss_compute_y` Function	Medium	Fixed
POL-06	Incomplete Conversion of `XY` Tokens to `LP` in the `compute_deposit` Function	Major	Fixed
POL-07	Administrator Privilege Allows Manipulation of Liquidity and Avoidance of Swap Fees	Major	Fixed
POL-08	Lack of Validation in `ss_compute_mint_amount_for_deposit` Function	Medium	Fixed

3 Participant Process

Here are the relevant actors with their respective abilities within the **`Suiswap`** Smart Contract:

Admin

- Admin can mint `TOKEN` tokens through the `increase_token_supply` function, and the upper limit of minting is `10000000000000000000`.
- Admin has the capability to mint and burn tokens, allowing them to issue or destroy tokens at will.
- Admin can manipulate the pool's exchange rate by changing the values of `pool.balance.bx` and `pool.balance.by` through the `do_change_basis function`, which will affect the actual amount of tokens received during the swap process.
- Admin can create a pool through the `create_pool` function.
- Admin can modify `admin_fee`, `lp_fee` and `th_fee` through `change_fee` function.
- Admin can freeze/unfreeze any pool, whether created by an admin or a user.

4 Findings

VPT-01 Unused Constant

Severity: Minor

Status: Fixed

Code Location: sources/vpt.move #L8, L12, L13; sources/sbalance.move#L13; sources/pool.move #L27, L29, L30, L52, L54, L64, L70, L76, L78, L96, L110, L112, L114

Descriptions: Certain variables declared in the contract are not referenced or utilized in any of the contract's functions or logic. These unused variables add unnecessary complexity to the codebase and can potentially confuse developers or auditors trying to understand the contract's functionality.

Suggestion: Unless there are specific plans for utilizing these variables in future updates or additions, it is advisable to remove them to improve code readability and maintainability.

Resolution: The client has followed our suggestion and fixed the issue.

VPT-02 Unnecessary Comparison of Boolean Values

Severity: Minor

Status: Fixed

Code Location: sources/vpt.move #L25; sources/pool.move #L681, L1033

Descriptions: The contract contains instances where boolean values are compared with `false` or `true` using conditional statements. This approach introduces unnecessary complexity and redundancy in the code, as boolean values can be directly utilized as conditions in if statements or loops.

Suggestion: It is recommended to follow best practices when working with boolean values and avoid unnecessary comparisons with `false` or `true`.

Resolution: The client has followed our suggestion and fixed the issue.

TKN-01 Lack of Event Logging in `do_withdraw_token_bank_admin_balance` Function

Severity: Minor

Status: Fixed

Code Location: sources/token.move #L736

Descriptions: The function `do_withdraw_token_bank_admin_balance` did not log an event when withdrawing the admin balance.

Suggestion: Add event logging for `do_withdraw_token_bank_admin_balance` function.

Resolution: The client has followed our suggestion and fixed the issue.

TKN-02 Lack of Permission Verification in `do_add_token_ido_whitelist` Function

Severity: Major

Status: Fixed

Code Location: sources/token.move #L936

Descriptions: The function `do_add_token_ido_whitelist` lacks permission verification, allowing anyone to add their address to the token IDO whitelist.

Suggestion: Designate specific addresses or roles as administrators who have the authority to modify the token IDO whitelist. Only authorized administrators should be able to invoke the `do_add_token_ido_whitelist` function.

Resolution: The client has followed our suggestion and fixed the issue.

TKN-03 Potential Event Bypass in `do_claim_token_airdrop_token_legacy` Function

Severity: Minor

Status: Fixed

Code Location: sources/token.move #L846

Descriptions: In `do_claim_token_airdrop_token`, it calls `do_claim_token_airdrop_token_legacy` and then emits the `ClaimTokenAirdropEvent` event.
However, `do_claim_token_airdrop_token_legacy` is public, developers can directly call `do_claim_token_airdrop_token_legacy` to skip event emitting.

Suggestion: Add event logging for `do_withdraw_token_bank_admin_balance` function.

Resolution: The client has followed our suggestion and fixed the issue.

TKN-04 Centralization Risk in `do_increase_token_supply` Function

Severity: Medium

Status: Partially Fixed

Code Location: sources/token.move #L721

Descriptions: The function `do_increase_token_supply` allows administrators to infinitely mint TOKEN tokens, posing a centralization risk.

Suggestion: Introduce a maximum supply limit for `TOKEN` tokens. Introduce a multi-signature approval mechanism for token minting.

Resolution: The client added a maximum coin minting limit of `10000000000000000000` to mitigate this issue.

TKN-05 Missing Separation of Production and Test Code for `token.move`

Severity: Major

Status: Fixed

Code Location: sources/token.move #L406-L422

Descriptions: The production code and test code for `token.move` is not properly differentiated and certain test functions are public and may be exploited. The test code needs to be removed or separated into its own `move` test code.

Suggestion: Extract the test code from the `token.move` file and place it in a separate file or directory specifically designated for tests. This separation ensures a clear distinction between the production and test code, making it easier to manage and maintain.

Resolution: The client has followed our suggestion and fixed the issue.

TKN-06 Add Assertion Validation for `share_minted`

Severity: Minor

Status: Fixed

Code Location: sources/token.move #L1003

Descriptions: The comment for the assertion `assert!(share_minted > 0, EComputationError);` can be removed, and additional validation for `share_minted` can be added.

Suggestion: Remove comment for assertion on returned `share_minted` value.

Resolution: The client has followed our suggestion and fixed the issue.

TKN-07 Assertion Error in `do_swap_x_to_y_direct` and `do_swap_x_to_y_direct` Functions

Severity: Medium

Status: Fixed

Code Location: sources/token.move #L1763, L1849

Descriptions: The function `do_swap_x_to_y_direct` has an assertion error and should be modified to `assert!(in_amount > 0 && vector::length(&cxs) > 0, EInvalidParameter);`. The same modification should be applied to the function `do_swap_x_to_y_direct`.

Suggestion: Update the assertion statement in both functions to include the condition `assert!(in_amount > 0 && vector::length(&cxs) > 0, EInvalidParameter);`.

Resolution: The client has followed our suggestion and fixed the issue.

TKN-08 Duplicate Code in `do_claim_token_airdrop_token_legacy` Function

Severity: Informational

Status: Fixed

Code Location: sources/token.move #L853, L860

Descriptions: There is duplicate code in the `do_claim_token_airdrop_token_legacy` function: `let sender = tx_context::sender(ctx);`.

Suggestion: Remove duplicate code.

Resolution: The client has followed our suggestion and fixed the issue.

TKN-09 Incorrect `token_type` Values in Recorded Events in `do_send_staked_token` Function

Severity: Minor

Status: Fixed

Code Location: sources/token.move #L798, L811

Descriptions: The `do_send_staked_token` function has incorrect `token_type` values in the recorded events. The `token_type` at line 798 should be `ESendTokenEvent_TokenType_StakedToken`, and the `token_type` at line 811 should be `ESendTokenEvent_TokenType_LinearUnlockStakedToken`.

Suggestion: Replace the incorrect `token_type` value at line 798 with `ESendTokenEvent_TokenType_StakedToken`. Replace the incorrect `token_type` value at line 811 with `ESendTokenEvent_TokenType_LinearUnlockStakedToken`.

Resolution: The client has followed our suggestion and fixed the issue.

TKN-10 Lack of Validation for `do_create_registry` and `do_create_token_farm` Functions

Severity: Minor

Status: Fixed

Code Location: sources/token.move #L1022

Descriptions: The function `do_create_registry` does not validate the parameter `boost_multiplier_data`. If the length of `boost_multiplier_data` is odd, it can lead to an abort in this function. Similarly, the function `do_create_token_farm` also has a similar issue.

Suggestion: By implementing proper validation for the `boost_multiplier_data` parameter, the solution enhances the robustness and reliability of the `do_create_registry` and `do_create_token_farm` functions, preventing potential aborts or failures caused by invalid inputs.

Resolution: The client has followed our suggestion and fixed the issue.

POL-01 Missing Emit Event

Severity: Minor

Status: Acknowledged

Code Location: sources/pool.move #L236–L241, L1595, L1676, L1688, L1694, L1700, L1716,

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

Suggestion: It is recommended to emit events for these functions.

POL–02 Missing Total Liquidity Restriction and Minimum Locking for `add_liquidity_direct_impl` Function

Severity: Major

Status: Fixed

Code Location: sources/pool.move #L1000

Descriptions: The function `add_liquidity_direct_impl` lacks the restriction for the total liquidity to be not less than 1000, and it also lacks the minimum liquidity locking requirement during the initial liquidity addition.

Suggestion: Add logic to the `add_liquidity_direct_impl` function to ensure that the total liquidity being added is not less than 1000. Before performing the liquidity addition, check whether the added liquidity meets the minimum requirement. If it is below 1000, reject the liquidity addition.

Resolution: The client has followed our suggestion and fixed the issue.

POL–03 The initialization value of `admin_fee` in the `do_create_registry` function is inconsistent with the comment

Severity: Informational

Status: Fixed

Code Location: sources/pool.move #L1625, L1627

Descriptions: The function `do_create_registry` has a discrepancy between the initialization value of `admin_fee` and its corresponding comment. The comment states that the value should be `0.03%`, but the actual initialization value is `2`.

Suggestion: Align initialization value with comment for `admin_fee`.

Resolution: The client has followed our suggestion and fixed the issue.

POL-04 Incorrect Calculation of `th_fee` Due to Deducting `admin_fee` from `balance` Before Computing `th_fee`

Severity: Minor

Status: Acknowledged

Code Location: sources/pool.move #L796

Descriptions: The function `collect_admin_and_th_fee_x` computes `admin_fee` and `th_fee` by first calculating the value of `admin_fee` and subtracting it from the `balance`. Then, it uses the reduced `balance` to compute `th_fee`. This leads to an underestimation of the calculated `th_fee`. The function `collect_admin_and_th_fee_y` also exhibits a similar issue.

Suggestion: First compute `th_fee` using the original `balance` without deducting `admin_fee`. After calculating `th_fee`, subtract the combined `admin_fee` and `th_fee` from the `balance` to obtain the updated `balance`.

Resolution: The client's response is that this calculation method will only cause a very small error, which can be ignored.

POL-05 Incorrect Loop Termination and Inconsistent Implementation in `ss_compute_y` Function

Severity: Medium

Status: Fixed

Code Location: sources/pool.move #L1441

Descriptions: The `ss_compute_y` function should abort instead of returning a value after the loop ends. It should iterate `255` times instead of `256`, which is inconsistent with the implementation of Curve.

Suggestion: Update the loop termination condition to ensure that it aborts after completing the desired 255 iterations. Remove the return statement after the loop, as the function should abort

instead of returning a value.

Resolution: The client has followed our suggestion and fixed the issue.

POL-06 Incomplete Conversion of `XY` Tokens to `LP` in the `compute_deposit` Function

Severity: Major

Status: Fixed

Code Location: sources/pool.move #L1171

Descriptions: The `compute_deposit` function may not always be able to fully convert the XY tokens provided by the user into LP tokens. It should handle returning any excess tokens.

Suggestion: Modify the `compute_deposit` function to include the necessary logic for returning excess tokens, ensuring the conversion process is complete and accurate.

Resolution: The client has followed our suggestion and fixed the issue.

POL-07 Administrator Privilege Allows Manipulation of Liquidity and Avoidance of Swap Fees

Severity: Major

Status: Fixed

Code Location: sources/pool.move #L933

Descriptions: When adding liquidity, the administrator can manipulate the price. They can first add one-sided liquidity X and then remove a portion of Y during liquidity removal, thereby avoiding swap fees. It is recommended to eliminate this privilege from the administrator.

Suggestion: Remove the privilege of the administrator to manipulate the price during liquidity addition.

Resolution: The client has followed our suggestion and fixed the issue.

POL-08 Lack of Validation in `ss_compute_mint_amount_for_deposit` Function

Severity: Medium

Status: Fixed

Code Location: sources/pool.move #L796

Descriptions: The function `ss_compute_mint_amount_for_deposit` lacks an assertion to verify that `d1` is greater than `d0`. When this function returns `0`, the external liquidity provider (LP) is not checked for being greater than `0`. This could lead to a situation where a user injects liquidity but receives an `LP` value of `0`.

Suggestion: Add an assertion within the `ss_compute_mint_amount_for_deposit` function to verify that `d1` is greater than `d0` before proceeding with further calculations.

Resolution: The client has followed our suggestion and fixed the 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.
- **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.