MobyDEX Smart Contract
Audit Report

Wed Aug 30 2023

contact@scalebit.xyz https://twitter.com/scalebit_

ScaleBit

MobyDEX Smart Contract Audit Report

1 Executive Summary

1.1 Project Information

Description	A launchpad and staking project
Type	Launchpad
Auditors	ScaleBit
Timeline	Mon Aug 21 2023 – Wed Aug 30 2023
Languages	Solidity
Platform	opBNB
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/mobydex-labs/mobydex-core
Commits	b9c86a5ea3587a966e35b6d67931421ef6c5a309 f25277096cce2ab5e49518655837683502d8b0bd 518a67794095d0b505eefd0ccde0fe968751329f

1.2 Files in Scope

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

ID	File	SHA-1 Hash
MOBYS	src/MOBYSale.sol	7763a9a82b7792b7f698d151f7dcd8825e844a6a
MAS	src/Masterchef.sol	7751510a36804d7fea51113ea97278668e306dfe
IMOBY	src/interfaces/IesMOBY.sol	9e5723fc2cbf8fa99ec5ea3681c7de3e7d92dfe8
MOBYS	src/MOBYSale.sol	fb3f866397eb2c58bf5b0a0ec0d322afdfd08968
IMOBY	src/interfaces/IesMOBY.sol	016b9e67fc296e925daab42e30e1bac2c10dd9c4
MOBYS	src/MOBYSale.sol	7cabde337653111da624db5e71750323959a6cef

1.3 Issue Statistic

Item	Count	Fixed	Acknowledged
Total	10	6	4
Informational	1	0	1
Minor	3	2	1
Medium	2	1	1
Major	4	3	1
Critical	0	0	0

1.4 ScaleBit Audit Breakdown

ScaleBit 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
- Number of rounding errors
- Unchecked External Call
- Unchecked CALL Return Values
- Functionality Checks
- Reentrancy
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic issues
- Gas usage
- Fallback function usage
- tx.origin authentication
- Replay attacks
- Coding style issues

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) 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 MobyDEX to identify any potential issues and vulnerabilities in the source code of the MobyDEX 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 10 issues of varying severity, listed below.

ID	Title	Severity	Status
MAS–1	Lack of Events Emit	Minor	Acknowledged
MAS–2	Unchecked Return Value	Medium	Fixed
MAS–3	Lack of Validation for Zero Address	Informational	Acknowledged
MAS–4	Centralization Risk	Major	Acknowledged
MAS–5	Incompatible with Deflation Tokens	Medium	Acknowledged
MOB–1	Unable to Claim Sale Token	Major	Fixed
MOB–2	Incorrect Conditional Judgment	Major	Fixed
MOB–3	Lack of Validation for `msg.value`	Major	Fixed
MOB–4	Unused State Variable	Minor	Fixed
MOB–5	Uncompilable Code	Minor	Fixed

3 Participant Process

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

Admin

- Admin can update the treasury address through `updateTreausry`.
- Admin can update the `rewardPerSec` through `updateRewardPerSec`.
- Admin can update the `esRewardPerSec` through `updateEsRewardPerSec`.
- Admin can update the `pool`, `rewardPerSec` and `esRewardPerSec` through `updateAndSetRewardPerSec`.
- Admin can update the `multiplier` through `updateMultiplier`.
- Admin can add a new `LP` to the pool through `add`.
- Admin can update the given pool's reward allocation point through `set`.
- Admin can set the start time through `setStartTime`.
- Admin can initialize the `MOBYSale` contract through `initialize`.
- Admin can set the amount of `offeringToken` through `setOfferingAmount`.
- Admin can set the amount of `lpToken` through `setRaisingAmount`.
- Admin can withdraw the `lpToken` in the `MOBYSale` contract through `withdrawAdmin` and `finalWithdraw`.
- Admin can withdraw the `offeringToken` in the `MOBYSale` contract through `finalWithdraw`.

User

- User can deposit the `LP` Toekn through `deposit`.
- User can withdraw the `LP` Token through `withdraw` and `emergencyWithdraw`.
- User can buy the `offeringToken` through `deposit`.
- User can claim the `offeringToken` and withdraw the `lpToken` through `harvest`.

4 Findings

MAS-1 Lack of Events Emit

Severity: Minor

Status: Acknowledged

Code Location:

src/Masterchef.sol#111,115,119,128,138,164,379

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 update functions.

MAS-2 Unchecked Return Value

Severity: Medium

Status: Fixed

Code Location:

src/Masterchef.sol#295

Descriptions:

The return value of the stake in the `transferPendingRewards` function is not checked and the `stake` function in the `IesMOBY` interface contract is different from the `esMOBY` contract, there is a return value in the `IesMOBY` interface contract but not in the `esMOBY` contract.

Suggestion:

It is recommended to check the return value in the `transferPendingRewards` function.

Resolution:

The client has followed our suggestion and fixed the issue.

MAS-3 Lack of Validation for Zero Address

Severity: Informational

Status: Acknowledged

Code Location:

src/Masterchef.sol#107

Descriptions:

There is no check for the zero address.

Suggestion:

It is recommended to add a check for the zero address.

MAS-4 Centralization Risk

Severity: Major

Status: Acknowledged

Code Location:

src/Masterchef.sol#107,111,115,119,128,138,164,379;

src/MOBYSale.sol#57,84,89,175,180

Descriptions:

There are some risks of centralization in the contract:

- Admin can update the treasury address through `updateTreausry`.
- Admin can update the `rewardPerSec` through `updateRewardPerSec`.
- Admin can update the `esRewardPerSec` through `updateEsRewardPerSec`.
- Admin can update the `pool`, `rewardPerSec` and `esRewardPerSec` through `updateAndSetRewardPerSec`.
- Admin can update the multiplier through `updateMultiplier`.
- Admin can add a new `LP` to the pool through `add`.
- Admin can update the given pool's reward allocation point through `set`.
- Admin can set the start time through `setStartTime`.
- Admin can initialize the `MOBYSale` contract through `initialize`.
- Admin can set the amount of `offeringToken` through `setOfferingAmount`.
- Admin can set the amount of `lpToken` through `setRaisingAmount`.
- Admin can withdraw the `lpToken` in the `MOBYSale` contract through `withdrawAdmin` and `finalWithdraw`.
- Admin can withdraw the `offeringToken` in the `MOBYSale` contract through `finalWithdraw`.

Suggestion:

It is recommended to take some measures to mitigate centralization risk.

MAS-5 Incompatible with Deflation Tokens

Severity: Medium

Status: Acknowledged

Code Location:

src/Masterchef.sol

Descriptions:

The MasterChef contracts do not appear to support rebasing/deflationary/inflationary tokens whose balance changes during transfers or over time.

Suggestion:

It is recommended to add the necessary checks including at least verifying the amount of tokens transferred to contracts before and after the actual transfer to infer any fees/interest.

MOB-1 Unable to Claim Sale Token

Severity: Major

Status: Fixed

Code Location:

src/MOBYSale.sol#97

Descriptions:

In the claim function, the variable `user.claimableAmount` is assigned a value of 0 at L106, and the user will not receive the Token when the function executes the transfer at L108.

Suggestion:

It is recommended to assigning the value of a variable to a temporary variable and then transfer the Token with the value of the temporary variable.

Resolution:

The client has followed our suggestion and fixed the issue.

MOB-2 Incorrect Conditional Judgment

Severity: Major

Status: Fixed

Code Location:

src/MOBYSale.sol#185

Descriptions:

In the `finalWithdraw` function, when `lpToken= address(0)`, the judgment condition is `_lpAmount > address(this).balance`, meaning that the amount withdrawn needs to be greater than the balance in the contract, which will result in the withdrawals never passing the conditional checks, resulting in the unsold `offeringToken` being locked in the contract. And `the_offerAmount < offeringToken.balanceOf(address(this))` will result in the unsold `offeringToken` can't be withdrawn completely.

Suggestion:

It is recommended to modify the judgment condition to `_lpAmount <= address(this).balance` and `_offerAmount <= offeringToken.balanceOf(address(this))`.

Resolution:

The client has followed our suggestion and fixed the issue.

MOB-3 Lack of Validation for `msg.value`

Severity: Major

Status: Fixed

Code Location:

src/MOBYSale.sol#94

Descriptions:

In the `deposit` function, it is supported to deposit `lpToken` and `ETH` when `address(lpToken) != address(0)`, the user can deposit both `lpToken` and `msg.value` at the same time, which will result in the deposited `ETH` lock in the contract and being unable to be withdrawn.

Suggestion:

It is recommended to add a check for `msg.value` in the `address(lpToken) ! = address(0)` condition, for example: `require(msg.value == 0, "need msg.value = 0")`;.

Resolution:

The client has followed our suggestion and fixed the issue.

MOB-4 Unused State Variable

Severity: Minor

Status: Fixed

Code Location:

src/MOBYSale.sol#42

Descriptions:

The variable adminClaimed is not used in the contract.

Suggestion:

It is recommended to remove the unused variable.

Resolution:

The client has followed our suggestion and fixed the issue.

MOB-5 Uncompilable Code

Severity: Minor

Status: Fixed

Code Location:

src/MOBYSale.sol#3

Descriptions:

Missing `;` after the version pragma caused the code not to compile.

Suggestion:

It is recommended to add `;` after the version pragma.

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