CERTIK

Mimo DeFi

Apr 18th, 2021

Table of Contents

Summary

Overview

- Project Summary
- Audit Summary
- Vulnerability Summary
- Audit Scope

Findings

- APC-01 : Missing Emit Events
- APV-01 : Missing Emit Events
- CPC-01 : Missing Parameter Check and Logical Issue
- CPV-01 : Missing Parameter Check and Logical Issue
- DMM-01 : `Check-effects-pattern` Not Used
- FDC-01 : Redundancy Data and Potentially Excessive Permissions
- FDV-01 : Redundancy Data and Potentially Excessive Permissions
- LMC-01 : Proper Usage of `public` and `external` Type
- MIM-01 : Discussion on Unknow Addresses and `airdrop` Function
- PFC-01 : Potentially Excessive Permissions
- PUA-01 : Discussion on Unknow Addresses and `airdrop` Function
- RVC-01 : Discussion on `repay` Function
- UCK-01 : Potentially Excessive Permissions on Upgrade
- VCC-01 : Missing Emit Event
- VDP-01 : Duplicated `require` Code
- VDV-01 : Duplicated `require` Code

Appendix

Disclaimer

About

Summary

This report has been prepared for Mimo DeFi smart contracts, to discover issues and vulnerabilities in the source code of their Smart Contract as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Static Analysis and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases given they are currently missing in the repository;
- Provide more comments per each function for readability, especially contracts are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

Overview

Project Summary

Project Name	Mimo DeFi
Platform	Ethereum
Language	Solidity
Codebase	
Commit	

Audit Summary

Delivery Date	Jun 16, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

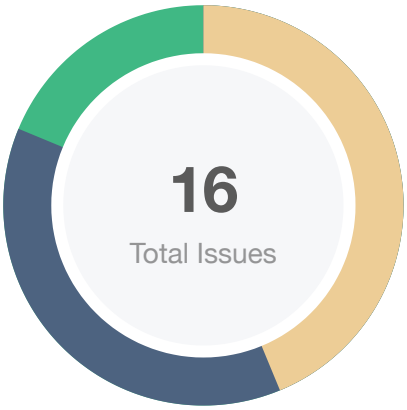
Vulnerability Summary

Total Issues	16
Critical	0
Major	0
Medium	0
Minor	7
Informational	6
Discussion	3

Audit Scope

ID	file	SHA256 Checksum

Findings



Critical	0 (0.00%)
Major	0 (0.00%)
Medium	0 (0.00%)
Minor	7 (43.75%)
Informational	6 (37.50%)
Discussion	3 (18.75%)

ID	Title	Category	Severity	Status
APC-01	Missing Emit Events	Control Flow	Informational	Pending
APV-01	Missing Emit Events	Control Flow	Informational	Pending
CPC-01	Missing Parameter Check and Logical Issue	Logical Issue	Minor	Pending
CPV-01	Missing Parameter Check and Logical Issue	Logical Issue	Minor	Pending
DMM-01	`Check-effects-pattern` Not Used	Logical Issue	Minor	Pending
FDC-01	Redundancy Data and Potentially Excessive Permissions	Logical Issue	Minor	Pending
FDV-01	Redundancy Data and Potentially Excessive Permissions	Logical Issue	Minor	Pending
LMC-01	Proper Usage of `public` and `external` Type	Gas Optimization	Informational	Pending
MIM-01	Discussion on Unknow Addresses and `airdrop` Function	Control Flow	Discussion	Pending
PFC-01	Potentially Excessive Permissions	Control Flow	Minor	Pending
PUA-01	Discussion on Unknow Addresses and `airdrop` Function	Control Flow	Discussion	Pending
RVC-01	Discussion on `repay` Function	Control Flow	Discussion	Pending
UCK-01	Potentially Excessive Permissions on Upgrade	Control Flow	Minor	Pending

ID	Title	Category	Severity	Status
VCC-01	Missing Emit Event	Control Flow	Informational	Pending
VDP-01	Duplicated `require` Code	Gas Optimization	Informational	Pending
VDV-01	Duplicated `require` Code	Gas Optimization	Informational	Pending

APC-01 | Missing Emit Events

Category	Severity	Location	Status
Control Flow	Informational	core/AddressProvider.sol: 17	Pending

Description

All of the functions in the `AddressProvider` contract are called by the owner. But these sensitive actions are defined without any event declarations.

Recommendation

We advise that add events for sensitive actions and emit them in the functions.

APV-01 | Missing Emit Events

Category	Severity	Location	Status
Control Flow	Informational	v1/AddressProviderV1.sol: 18	Pending

Description

All of the functions in the `AddressProviderV1` contract are called by the owner. But these sensitive actions are defined without any event declarations.

Recommendation

We advise that add events for sensitive actions and emit them in the functions.

CPC-01 | Missing Parameter Check and Logical Issue

Category	Severity	Location	Status
Logical Issue	Minor	core/ConfigProvider.sol: 124~129	Pending

Description

If the removed collateral is the lastest one, it does not need to move last entry forward.

And the line 125 code may be error. It should update the `numCollateralConfigs` entry, instead of the `collateralIds[_collateralConfigs[id].collateralType]` entry. Do more testing on this point.

```
uint256 id = collateralIds[_collateralType];
......
_collateralConfigs[id] = _collateralConfigs[numCollateralConfigs];
collateralIds[_collateralConfigs[id].collateralType] = id;
```

Recommendation

Refer to change the code as the following example:

```
......
if (id < numCollateralConfigs) {
    _collateralConfigs[id] = _collateralConfigs[numCollateralConfigs]; // move last entry
forward
    collateralIds[_collateralConfigs[numCollateralConfigs].collateralType] = id; //
update id for last entry
}
//delete the last entry
```

CPV-01 | Missing Parameter Check and Logical Issue

Category	Severity	Location	Status
Logical Issue	Minor	v1/ConfigProviderV1.sol: 91~96	Pending

Description

If the removed collateral is the lastest one, it does not need to move last entry forward.

And the line 94 code may be error. It should update the `numCollateralConfigs` entry, instead of the `collateralIds[_collateralConfigs[id].collateralType]` entry. Do more testing on this point.

```
uint256 id = collateralIds[_collateralType];
.......
_collateralConfigs[id] = _collateralConfigs[numCollateralConfigs];
collateralIds[_collateralConfigs[id].collateralType] = id;
```

Recommendation

Refer to change the code as the following example:

```
.......
if (id < numCollateralConfigs) {
    _collateralConfigs[id] = _collateralConfigs[numCollateralConfigs]; // move last entry
forward
    collateralIds[_collateralConfigs[numCollateralConfigs].collateralType] = id; //
update id for last entry
}
//delete the last entry
```

DMM-01 | `Check-effects-pattern` Not Used

Category	Severity	Location	Status
Logical Issue	Minor	liquidityMining/DemandMiner.sol: 40~41	Pending

Description

During `withdraw` function calls state variables for balance are changed after transfer are done. This will lead to reentrancy issue.

Recommendation

It is recommended to follow checks-effects-interactions pattern for cases like this. It shields public functions from re-entrancy attacks. It's always a good practice to follow this pattern. `checks-effects-interactions` pattern also applies to ERC20 tokens as they can inform the recipient of a transfer in certain implementations.

Refer https://docs.soliditylang.org/en/develop/security-considerations.html?highlight=check-effects%23use-the-checks-effects-interactions-pattern

FDC-01 | Redundancy Data and Potentially Excessive Permissions

Category	Severity	Location	Status
Logical Issue	Minor	fees/FeeDistributor.sol: 69~72	Pending

Description

In the `for` loop this project will remove apart of `shares`. At the same time, the code will remove all of the `payees`. Why don't this project remove all of the legacy `shares`? If it's acceptable, we advise that remove all of the `shares` and `payees` at the same time.

The `changePayees` function is only called by the owner, and it allows the caller to remove all of the `payees` and `shares` data. To improve the trustworthiness of this project, any plan to change on the assets of the `payees` and `shares` should move to the execution queue of the `Timelock`, and also add an `emit event`, and make the owner Multi-sig.

Recommendation

We advise that remove all of the `payees` and `shares` data at the same time, and add an `emit event` at the `changePayees` function. And it should transfer the owner of this contract to `Timelock`, and make the owner Multi-sign. It is better to add community voting for `changePayees` function.

FDV-01 | Redundancy Data and Potentially Excessive Permissions

Category	Severity	Location	Status
Logical Issue	Minor	v1/FeeDistributorV1.sol: 105~108	Pending

Description

In the `for` loop this project will remove apart of `shares`. At the same time, the code will remove all of the `payees`. Why don't this project remove all of the legacy `shares`? If it's acceptable, we advise that remove all of the `shares` and `payees` at the same time.

The `changePayees` function is only called by the owner, and it allows the caller to remove all of the `payees` and `shares` data. To improve the trustworthiness of this project, any plan to change on the assets of the `payees` and `shares` should move to the execution queue of the `Timelock`, and also add an `emit event`, and make the owner Multi-sig.

Recommendation

We advise that remove all of the `payees` and `shares` data at the same time, and add an `emit event` at the `changePayees` function. And it should transfer the owner of this contract to `Timelock`, and make the owner Multi-sign. It is better to add community voting for `changePayees` function.

LMC-01 | Proper Usage of `public` and `external` Type

Category	Severity	Location	Status
Gas Optimization	Informational	core/LiquidiationManager.sol: 36	Pending

Description

The `public` functions that are never called by the contract could be declared `external`. When the inputs are arrays `external` functions are more efficient than `public` functions. And the same issue on the other functions.

Recommendation

We advise that use the `external` attribute for the functions never called by the contract.

MIM-01 | Discussion on Unknow Addresses and `airdrop` Function

Category	Severity	Location	Status
Control Flow	Discussion	liquidityMining/MIMODistributor.sol: 82~85	Pending

Description

The `airdrop` function in `PreUseAirdrop` contract will mint lots of MIMO token to the `0x002F042Dc7622cD8426457df28525692B3CaCc5E` and `0x71fE6c4abAfEF47CF23C4b9fB45f7fcBc238d624` addresses.

What are the two addresses? When will the `airdrop` function be called?

PFC-01 | Potentially Excessive Permissions

Category	Severity	Location	Status
Control Flow	Minor	core/PriceFeed.sol: 43~48, 54~58	Pending

Description

The `setAssetOracle` and `setEurOracle` functions are only called by the owner, and they allow the caller to set the oracle for the given asset and the oracle for EUR.

To improve the trustworthiness of the project, any plan to set the mission should move to the execution queue of the `Timelock` and also add an `emit event`, or make the owner Multi-sig.

Recommendation

We advise that add an `emit event` at the `setAssetOracle` and `setEurOracle` functions. And transfer the owner of this contract to `Timelock`, or make the owner Multi-sig. It's better to add community voting.

PUA-01 | Discussion on Unknow Addresses and `airdrop` Function

Category	Severity	Location	Status
Control Flow	Discussion	upgrade/PreUseAirdrop.sol: 34~35	Pending

Description

The `airdrop` function in `PreUseAirdrop` contract will mint lots of MIMO token to the `0x002F042Dc7622cD8426457df28525692B3CaCc5E` and `0x71fE6c4abAfEF47CF23C4b9fB45f7fcBc238d624` addresses.

What are the two addresses? When will the `airdrop` function be called?

RVC-01 | Discussion on `repay` Function

Category	Severity	Location	Status
Control Flow	Discussion	upgrade/RepayVault.sol: 32	Pending

Description

The `repay` function is only called by owner. It allows the caller to repay 10 tokens to all of the different vaults. And then it will transfer all of the `PAR` asset of this contract to the caller. When will the owner call this `repay` function?

To improve the trustworthiness of the project, any plan to transfer assets should move to the execution queue of the `Timelock`, and also add an `emit event`, or make the owner Multi-sig.

Recommendation

We advise that add an `emit event` at the `repay` function. And it should transfer the owner of the contract to `Timelock`, and make the owner Multi-sig. It is better to add community voting.

UCK-01 | Potentially Excessive Permissions on Upgrade

Category	Severity	Location	Status
Control Flow	Minor	upgrade/Upgrade.sol: 70	Pending

Description

The `Upgrade.upgrade`, `VaultsCore.upgrade`, `VaultsCoreV1.upgrade` functions are only called by the owner, and they allow the caller to transfer the assets of current version contracts to new version contracts.

To improve the trustworthiness of the project, any plan to set the mission should move to the execution queue of the `Timelock` and also add an `emit event`, or make the owner Multi-sig.

Recommendation

We advise that add an `emit event` at these functions. And transfer the owner of this contract to `Timelock`, or make the owner Multi-sig. It's better to add community voting.

VCC-01 | Missing Emit Event

Category	Severity	Location	Status
Control Flow	Informational	core/VaultsCore.sol: 96~99	Pending

Description

The sensitive action is defined without event declarations.

Recommendation

We advise that add an event for this sensitive action, and emit it in this function.

VDP-01 | Duplicated `require` Code

Category	Severity	Location	Status
Gas Optimization	Informational	core/VaultsDataProvider.sol: 42	Pending

Description

The line 39 `require` code is same as the line 42 code.

Recommendation

We advise that remove the redundancy code.

VDV-01 | Duplicated `require` Code

Category	Severity	Location	Status
Gas Optimization	Informational	v1/VaultsDataProviderV1.sol: 42	Pending

Description

The line 39 `require` code is same as the line 42 code.

Recommendation

We advise that remove the redundancy code.

Appendix

Finding Categories

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how block.timestamp works.

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functions being invoke-able by anyone under certain circumstances.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to the Company in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes without CertiK's prior written consent.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts CertiK to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. CertiK's position is that each company and individual are responsible for their own due diligence and continuous security. CertiK's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

About

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we're able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.