CERTIK

Security Assessment

MCDEX DAO

Apr 15th, 2021

Summary

This report has been prepared for MCDEX DAO 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 Dynamic Analysis, 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	MCDEX DAO
Platform	Ethereum
Language	Solidity
Codebase	https://github.com/mcarloai/mcdex3-governance
Commits	fd91f531439e3e3c0099fd3a586fba625b85172b

Audit Summary

Delivery Date	Apr 15, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

Vulnerability Summary

Total Issues	23
● Critical	0
● Major	3
● Minor	9
● Informational	11
● Discussion	0

Audit Scope

ID	file	SHA256 Checksum
ACK	contracts/Authenticator.sol	fe0a6fbb69ae8b95b680aca2fb5bfc774c763f528e7d8a4ab0937ac48c5fc67a
BBC	contracts/BalanceBroadcaster.sol	911889bb3a35c6a103d2e6a941f8df7844b61aeb0580d517033425fd9f6f5325
CCK	contracts/Comp.sol	77b8b3aa453065bd2811f9a2a7085bebe2cf3ec81bbf75860b1a7ec7266e0c6c
DEC	contracts/DataExchange.sol	2634494196f422bde7b91aad1ed49ea797e1b026ca8d4df2ffc2ad1c4a986ce4
GAC	contracts/GovernorAlpha.sol	dc183c5944aca32040c92895921850bbf897cd3928ec4fb47afe7361a339a592
MIC	contracts/MintInitiator.sol	7ff41a07297cbdeb73514288bb3eadb55b7f360d634c96525046db4b8b37d1a9
MCK	contracts/Minter.sol	752b0a5a3a24180839dcac73173fc0d39885d37f6c4df79ea701ba7519ebcac1
TCK	contracts/Timelock.sol	73298e89dc378cfa026401cba506287da45afec9a3276da3fef9408a7d02d243
VCC	contracts/ValueCapture.sol	614d03db5cd74ac9e2bcda33b56fb90d035a66be00d832bac2b73ac7560af886
VCK	contracts/Vault.sol	0745ad1d590e490cfa59a66f9e32016bffb03bf3c299e1d8122c20652f7d34c9
XMC	contracts/XMCB.sol	c2e39136614ec84f385a9e2c6f1c1fe617070c8fc7fd87030aea1d397099fe27
RDC	contracts/components/staking/RewardDistribution.sol	46bdfe6adf66f75b8b8de8769488fa5a9d80175876bd700f096a057c7a41da0d
TCC	contracts/libraries/TokenConversion.sol	10c669ba6da35ac0bc0adf659f5cf62b6ce4d1b71ceeceb98845b549f14a467c

Centralized Risks

Additionally, to bridge the trust gap between the administrator and users, the administrator needs to express a sincere attitude with the consideration of the administrator team’s anonymousness. The administrator has the responsibility to notify users with the following capability of the administrator:

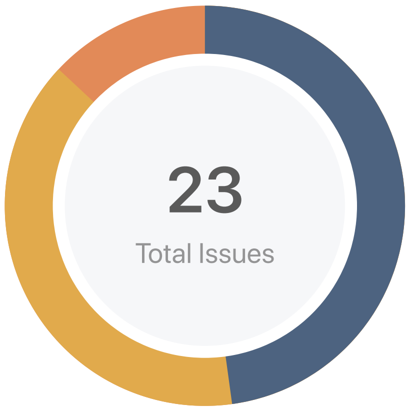
- Administrators can send assets to any address via "Vault.transferETH()", "Vault.transferERC20()" and "Vault.transferERC721()" functions.

To improve the trustworthiness of the project, any dynamic runtime updates in the project should be notified to the community. Any plan to invoke the abovementioned functions should be also considered to move to the execution queue of the Timelock contract.

Financial Models

Financial models of blockchain protocols need to be resilient to attacks. It needs to pass simulations and verifications to guarantee the security of the overall protocol.

Findings



🟥	Critical	0 (0.00%)
🟧	Major	3 (13.04%)
🟨	Minor	9 (39.13%)
🟦	Informational	11 (47.83%)
🟩	Discussion	0 (0.00%)

ID	Title	Category	Severity	Status
ACK-01	Proper usage of "public" and "external" type	Gas Optimization	● Informational	Partially Resolved
BBC-01	Proper usage of "public" and "external" type	Gas Optimization	● Informational	Partially Resolved
CCK-01	Proper usage of "public" and "external" type	Gas Optimization	● Informational	Partially Resolved
DEC-01	Proper usage of "public" and "external" type	Gas Optimization	● Informational	Partially Resolved
DEC-02	Function state mutability can be restricted to `view`	Gas Optimization	● Informational	✓ Resolved
DEC-03	Logic Issue in Modifier `onlyL1()` and `onlyL2()`	Logical Issue	● Informational	✓ Resolved
DEC-04	Contracts that lock Ether	Logical Issue	● Major	✓ Resolved
GAC-01	Uninitialized Constant `DATA_EXCHANGE_ADDRESS`	Logical Issue	● Informational	ⓘ Acknowledged
MCK-01	Missing Important Checks in Function `setDevAccount`	Logical Issue	● Minor	✓ Resolved
MCK-02	Logical Issue _mint	Logical Issue	● Minor	✓ Resolved
MCK-03	Uninitialized Variable `seriesALastUpdateBlock`	Logical Issue	● Minor	✓ Resolved

ID	Title	Category	Severity	Status
MCK-04	Logical Issue of `_updateExtraMintableAmount`	Logical Issue	● Minor	✓ Resolved
MCK-05	Logical Issue of `_getBaseMintableAmount`	Logical Issue	● Minor	✓ Resolved
MIC-01	Unused Constant `MCB_MINTER_ADDRESS`	Logical Issue	● Informational	✓ Resolved
RDC-01	Proper usage of "public" and "external" type	Gas Optimization	● Informational	Partially Resolved
RDC-02	File not found	Compiler Error	● Major	✓ Resolved
RDC-03	Logical Issue in `notifyRewardAmount()`	Logical Issue	● Minor	ⓘ Acknowledged
VCC-01	Tautology or Contradiction	Gas Optimization	● Informational	✓ Resolved
VCC-02	Functions That Need Permission Control in `ValueCapture`	Logical Issue	● Minor	⊗ Declined
VCC-03	Missing Checks in `setConvertor()`	Logical Issue	● Minor	✓ Resolved
VCK-01	**Centralized Risk**	**Centralization / Privilege**	● **Major**	✓ **Resolved**
VCK-02	Missing Important Checks on some transfer functions	Logical Issue	● Minor	✓ Resolved
XMC-01	Proper usage of "public" and "external" type	Gas Optimization	● Informational	Partially Resolved

ACK-01 | Proper usage of "public" and "external" type

Category	Severity	Location	Status
Gas Optimization	● Informational	contracts/Authenticator.sol: 27	Partially Resolved

Description

"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.

Examples:

Functions `Authenticator.hasRoleOrAdmin()`, `BalanceBroadcaster.componentCount()`, `BalanceBroadcaster.isComponent()`, `BalanceBroadcaster.listComponents()`, `DataExchange.getData()`,`DataExchange.getDataLastUpdateTimestamp()` , `Comp.totalSupply()`, `Comp.delegate()`, `Comp.delegateBySig()`, `Comp.getPriorVotes()`, `XMCB.rawBalanceOf()`, `XMCB.rawTotalSupply()` , `RewardDistribution.baseToken()` , `RewardDistribution.createRewardPlan()`, `RewardDistribution.getRewardTokens()`, `RewardDistribution.getRewardPlan()`, `RewardDistribution.setRewardRate()`, `RewardDistribution.notifyRewardAmount()`, `RewardDistribution.getAllRewards()` on the aforementioned lines.

Recommendation

Consider using the "external" attribute for functions never called from the contract.

Alleviation

The team heeded our advice and resolved this issue in commit 322d33cc3ed0b7efee05286949a91ec4d1a68974.

BBC-01 | Proper usage of "public" and "external" type

Category	Severity	Location	Status
Gas Optimization	● Informational	contracts/BalanceBroadcaster.sol: 25, 32, 39	Partially Resolved

Description

"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.

Examples:

Functions `Authenticator.hasRoleOrAdmin()`, `BalanceBroadcaster.componentCount()`, `BalanceBroadcaster.isComponent()`, `BalanceBroadcaster.listComponents()`, `DataExchange.getData()`,`DataExchange.getDataLastUpdateTimestamp()` , `Comp.totalSupply()`, `Comp.delegate()`, `Comp.delegateBySig()`, `Comp.getPriorVotes()`, `XMCB.rawBalanceOf()`, `XMCB.rawTotalSupply()` , `RewardDistribution.baseToken()` , `RewardDistribution.createRewardPlan()`, `RewardDistribution.getRewardTokens()`, `RewardDistribution.getRewardPlan()`, `RewardDistribution.setRewardRate()`, `RewardDistribution.notifyRewardAmount()`, `RewardDistribution.getAllRewards()` on the aforementioned lines.

Recommendation

Consider using the "external" attribute for functions never called from the contract.

Alleviation

The team heeded our advice and resolved this issue in commit 322d33cc3ed0b7efee05286949a91ec4d1a68974.

CCK-01 | Proper usage of "public" and "external" type

Category	Severity	Location	Status
Gas Optimization	● Informational	contracts/Comp.sol: 70, 121, 134, 172	Partially Resolved

Description

"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.

Examples:

Functions `Authenticator.hasRoleOrAdmin()`, `BalanceBroadcaster.componentCount()`, `BalanceBroadcaster.isComponent()`, `BalanceBroadcaster.listComponents()`, `DataExchange.getData()`,`DataExchange.getDataLastUpdateTimestamp()` , `Comp.totalSupply()`, `Comp.delegate()`, `Comp.delegateBySig()`, `Comp.getPriorVotes()`, `XMCB.rawBalanceOf()`, `XMCB.rawTotalSupply()` , `RewardDistribution.baseToken()` , `RewardDistribution.createRewardPlan()`, `RewardDistribution.getRewardTokens()`, `RewardDistribution.getRewardPlan()`, `RewardDistribution.setRewardRate()`, `RewardDistribution.notifyRewardAmount()`, `RewardDistribution.getAllRewards()` on the aforementioned lines.

Recommendation

Consider using the "external" attribute for functions never called from the contract.

Alleviation

The team heeded our advice and resolved this issue in commit 322d33cc3ed0b7efee05286949a91ec4d1a68974.

DEC-01 | Proper usage of "public" and "external" type

Category	Severity	Location	Status
Gas Optimization	● Informational	contracts/DataExchange.sol: 79, 86	Partially Resolved

Description

"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.

Examples:

Functions `Authenticator.hasRoleOrAdmin()`, `BalanceBroadcaster.componentCount()`, `BalanceBroadcaster.isComponent()`, `BalanceBroadcaster.listComponents()`, `DataExchange.getData()`,`DataExchange.getDataLastUpdateTimestamp()` , `Comp.totalSupply()`, `Comp.delegate()`, `Comp.delegateBySig()`, `Comp.getPriorVotes()`, `XMCB.rawBalanceOf()`, `XMCB.rawTotalSupply()` , `RewardDistribution.baseToken()` , `RewardDistribution.createRewardPlan()`, `RewardDistribution.getRewardTokens()`, `RewardDistribution.getRewardPlan()`, `RewardDistribution.setRewardRate()`, `RewardDistribution.notifyRewardAmount()`, `RewardDistribution.getAllRewards()` on the aforementioned lines.

Recommendation

Consider using the "external" attribute for functions never called from the contract.

Alleviation

The team heeded our advice and resolved this issue in commit 322d33cc3ed0b7efee05286949a91ec4d1a68974.

DEC-02 | Function state mutability can be restricted to `view`

Category	Severity	Location	Status
Gas Optimization	● Informational	contracts/DataExchange.sol: 291	✓ Resolved

Description

Functions on the afore-mentioned lines can be restricted to `view`.

Recommendation

Consider to restrict the functions to `view`.

Alleviation

The team heeded our advice and resolved this issue in commit 322d33cc3ed0b7efee05286949a91ec4d1a68974.

DEC-03 | Logic Issue in Modifier `onlyL1()` and `onlyL2()`

Category	Severity	Location	Status
Logical Issue	● Informational	contracts/DataExchange.sol: 47~55(DataExchange)	✓ Resolved

Description

Two functions have the same `require` with very different error messages. The modifier `onlyL2()` maybe require `_isL2Net()`.

Recommendation

Considier checking the logic and fixing it.

Alleviation

The team heeded our advice and resolved this issue in commit 322d33cc3ed0b7efee05286949a91ec4d1a68974.

DEC-04 | Contracts that lock Ether

Category	Severity	Location	Status
Logical Issue	● Major	contracts/DataExchange.sol: 37	✓ Resolved

Description

`DataExchange` contract does not have a function to withdraw the ether, every ether sent to it will be lost. Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#contracts-that-lock-ether

Recommendation

Consider removing the payable attribute or add a withdraw function.

Alleviation

The team heeded our advice and resolved this issue in commit 322d33cc3ed0b7efee05286949a91ec4d1a68974.

GAC-01 | Uninitialized Constant `DATA_EXCHANGE_ADDRESS`

Category	Severity	Location	Status
Logical Issue	● Informational	contracts/GovernorAlpha.sol: 9(GovernorAlpha)	ⓘ Acknowledged

Description

The constant `DATA_EXCHANGE_ADDRESS` is set to address(0) currently, please confirm it is set correctly later before deployment.

Recommendation

Consider setting the constant to correct address.

MCK-01 | Missing Important Checks in Function `setDevAccount`

Category	Severity	Location	Status
Logical Issue	● Minor	contracts/Minter.sol: 127	✓ Resolved

Description

Functions `setDevAccount()` on the afore-mentioned lines are missing parameter validations.

If these functions were called by mistake, there is no way to recover.

Recommendation

Consider adding below checks:

```
require(devAccount_ != address (0x0), "zero address now allowed");
```

Alleviation

The team heeded our advice and resolved this issue in commit
322d33cc3ed0b7efee05286949a91ec4d1a68974.

MCK-02 | Logical Issue _mint

Category	Severity	Location	Status
Logical Issue	● Minor	contracts/Minter.sol: 314	✓ Resolved

Description

The checks of totalSupply is better to be moved before mint.

Alleviation

The team heeded our advice and resolved this issue in commit 322d33cc3ed0b7efee05286949a91ec4d1a68974.

MCK-03 | Uninitialized Variable `seriesALastUpdateBlock`

Category	Severity	Location	Status
Logical Issue	● Minor	contracts/Minter.sol: 156~171(Minter)	✓ Resolved

Description

Variable `seriesALastUpdateBlock` has value 0 when not initialized.

Recommendation

Consider using function `_getSeriesALastUpdateBlock()` to get the value instead.

Alleviation

The team heeded our advice and resolved this issue in commit 322d33cc3ed0b7efee05286949a91ec4d1a68974.

MCK-04 | Logical Issue of `_updateExtraMintableAmount`

Category	Severity	Location	Status
Logical Issue	● Minor	contracts/Minter.sol: 138	✓ Resolved

Description

`extraMintableAmount` should be accumulated.

```
extraMintableAmount = incrementalCapturedValue.sub(baseMintableAmount);
```

Recommendation

Consider changing as below example:

```
extraMintableAmount += incrementalCapturedValue.sub(baseMintableAmount);
```

Alleviation

The team heeded our advice and resolved this issue in commit 0f5fc9eff1a4bf9ff404c379c0bb9dee77a2b349.

MCK-05 | Logical Issue of `_getBaseMintableAmount`

Category	Severity	Location	Status
Logical Issue	● Minor	contracts/Minter.sol: 331	✓ Resolved

Description

Function `_getBaseMintableAmount()` returns a value contains `extraMintableAmount` but minting base part only records `baseMintedAmount` without decresing `extraMintableAmount`.

Alleviation

The team heeded our advice and resolved this issue in commit 5fc04a894bd1f4c5e75bac233b3d1c1b33c664e2. `ExtraMintableAmount` will be deducted when in absence of `baseMintableAmount` now. The team also added a variable `baseMintable` to record these inputs and outputs, replacing the previous algorithm `rate * blocks + extraMintableAmount`.

MIC-01 | Unused Constant `MCB_MINTER_ADDRESS`

Category	Severity	Location	Status
Logical Issue	● Informational	contracts/MintInitiator.sol: 23(MintInitiator)	✓ Resolved

Description

Constant `MCB_MINTER_ADDRESS` is declared but never used and it has the same value with `ARB_SYS_ADDRESS`.

Recommendation

Consider removing the variable.

Alleviation

The team heeded our advice and resolved this issue in commit 322d33cc3ed0b7efee05286949a91ec4d1a68974.

RDC-01 | Proper usage of "public" and "external" type

Category	Severity	Location	Status
Gas Optimization	● Informational	contracts/components/staking/RewardDistribution.sol: 79~95, 121, 134, 146, 157, 185, 265	Partially Resolved

Description

"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.

Examples:

Functions `Authenticator.hasRoleOrAdmin()`, `BalanceBroadcaster.componentCount()`, `BalanceBroadcaster.isComponent()`, `BalanceBroadcaster.listComponents()`, `DataExchange.getData()`,`DataExchange.getDataLastUpdateTimestamp()` , `Comp.totalSupply()`, `Comp.delegate()`, `Comp.delegateBySig()`, `Comp.getPriorVotes()`, `XMCB.rawBalanceOf()`, `XMCB.rawTotalSupply()` , `RewardDistribution.baseToken()` , `RewardDistribution.createRewardPlan()`, `RewardDistribution.getRewardTokens()`, `RewardDistribution.getRewardPlan()`, `RewardDistribution.setRewardRate()`, `RewardDistribution.notifyRewardAmount()`, `RewardDistribution.getAllRewards()` on the aforementioned lines.

Recommendation

Consider using the "external" attribute for functions never called from the contract.

Alleviation

The team heeded our advice and resolved this issue in commit 322d33cc3ed0b7efee05286949a91ec4d1a68974.

RDC-02 | File not found

Category	Severity	Location	Status
Compiler Error	● Major	contracts/components/staking/RewardDistribution.sol: 13, 13	✓ Resolved

Description

The imported file is not found.

```
import "hardhat/console.sol";
```

Recommendation

Consider fixing the compiler error.

Alleviation

The team heeded our advice and resolved this issue in commit 322d33cc3ed0b7efee05286949a91ec4d1a68974.

RDC-03 | Logical Issue in `notifyRewardAmount()`

Category	Severity	Location	Status
Logical Issue	● Minor	contracts/components/staking/RewardDistribution.sol: 185~205	ⓘ Acknowledged

Description

Ensure to transfer enough reward token to current contract, otherwise `getReward()` will fail for not enough rewards.

Recommendation

Consider ensuring enough reward tokens in current contract.

Alleviation

The MCDEX team stating "This issue is existing. Due to the special minting mechanism, they will fix the issue in their own timeframe".

VCC-01 | Tautology or Contradiction

Category	Severity	Location	Status
Gas Optimization	● Informational	contracts/ValueCapture.sol: 120	✓ Resolved

Description

Function `addUSDToken()` contains a tautology or contradiction:

```
require(decimals >= 0 && decimals <= 18, "decimals out of range");
```

unit is always greater than 0.

Recommendation

Consider to change the codes as below example:

```
require(decimals <= 18, "decimals out of range");
```

Alleviation

The team heeded our advice and resolved this issue in commit 322d33cc3ed0b7efee05286949a91ec4d1a68974.

VCC-02 | Functions That Need Permission Control in `ValueCapture`

Category	Severity	Location	Status
Logical Issue	● Minor	contracts/ValueCapture.sol: 181~221(ValueCapture)	⊗ Declined

Description

Function `forwardAsset()` and `feedCaptureValueToL1()` need modifier `onlyAuthorized` to restrict access permissions.

Alleviation

The recommendation was not taken into account, with the MCDEX team stating "Function `forwardAsset()` forwards the USD tokens to the vault within the slippage tolerance limit. Function `feedCaptureValueToL1()` decides the max `mintable` amount. The minting actions will be executed by Timelock based on voting results. So we think there is no need to restrict access permissions on these two functions."

VCC-03 | Missing Checks in `setConvertor()`

Category	Severity	Location	Status
Logical Issue	● Minor	contracts/ValueCapture.sol: 158~174(ValueCapture), 248~268(ValueCapture)	Resolved

Description

Function `setConverter()` lacks checks on `tokenIn`. It is necessary to check if `tokenIn == converter_.tokenIn()` and it is also recommended to add this checking in `_convertTokenToUSD()`.

Besides, check the oracle as well if you can.

Recommendation

Consider adding checks in the functions mentioned.

Alleviation

The team heeded our advice and resolved this issue in commit 322d33cc3ed0b7efee05286949a91ec4d1a68974.

VCK-01 | Centralized Risk

Category	Severity	Location	Status
Centralization / Privilege	● **Major**	**contracts/Vault.sol: 66**	✓ **Resolved**

Description

To bridge the trust gap between the administrator and users, the administrator needs to express a sincere attitude with the consideration of the administrator team's anonymousness. The administrator has the responsibility to notify users with the following capability of the administrator:

- Administrators can send assets to any address via "Vault.transferETH()", "Vault.transferERC20()" and "Vault.transferERC721()" functions.

Recommendation

To improve the trustworthiness of the project, any dynamic runtime updates in the project should be notified to the community. Any plan to invoke the above-mentioned functions should be also considered to move to the execution queue of the Timelock contract.

Alleviation

The team heeded our advice and resolved this issue in commit 322d33cc3ed0b7efee05286949a91ec4d1a68974. The `VAULT_ADMIN_ROLE` role has been removed. Hence only `DEFAULT_ADMIN_ROLE` role can do the transfers. `DEFAULT_ADMIN_ROLE` will be transferred to timelock contract.

VCK-02 | Missing Important Checks on some transfer functions

Category	Severity	Location	Status
Logical Issue	● Minor	contracts/Vault.sol: 67, 84, 97	✓ Resolved

Description

Functions `transferETH()`, `transferERC20()`, `transferERC721()` on the afore-mentioned lines are missing parameter validations.

Recommendation

Consider adding below checks:

```
require(recipient != address (0x0), "recipient is address zero");
```

Alleviation

The team heeded our advice and resolved this issue in commit 322d33cc3ed0b7efee05286949a91ec4d1a68974.

XMC-01 | Proper usage of "public" and "external" type

Category	Severity	Location	Status
Gas Optimization	● Informational	contracts/XMCB.sol: 88, 95	Partially Resolved

Description

"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.

Examples:

Functions `Authenticator.hasRoleOrAdmin()`, `BalanceBroadcaster.componentCount()`, `BalanceBroadcaster.isComponent()`, `BalanceBroadcaster.listComponents()`, `DataExchange.getData()`,`DataExchange.getDataLastUpdateTimestamp()` , `Comp.totalSupply()`, `Comp.delegate()`, `Comp.delegateBySig()`, `Comp.getPriorVotes()`, `XMCB.rawBalanceOf()`, `XMCB.rawTotalSupply()` , `RewardDistribution.baseToken()` , `RewardDistribution.createRewardPlan()`, `RewardDistribution.getRewardTokens()`, `RewardDistribution.getRewardPlan()`, `RewardDistribution.setRewardRate()`, `RewardDistribution.notifyRewardAmount()`, `RewardDistribution.getAllRewards()` on the aforementioned lines.

Recommendation

Consider using the "external" attribute for functions never called from the contract.

Alleviation

The team heeded our advice and resolved this issue in commit 322d33cc3ed0b7efee05286949a91ec4d1a68974.

Appendix

Finding Categories

Gas Optimization

Gas Optimization findings refer to exhibits that 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.

Mathematical Operations

Mathematical Operation exhibits entail findings that relate to mishandling of math formulas, such as overflows, incorrect operations etc.

Logical Issue

Logical Issue findings are exhibits that 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.

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.

Data Flow

Data Flow findings describe faults in the way data is handled at rest and in memory, such as the result of a struct assignment operation affecting an in-memory struct rather than an in storage one.

Language Specific

Language Specific findings are issues that would only arise within Solidity, i.e. incorrect usage of private or delete .

Coding Style

Coding Style findings usually do not affect the generated byte-code and comment on how to make the codebase more legible and as a result easily maintainable.

Inconsistency

Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code, such as a constructor assignment imposing different require statements on the input variables than a setter function.

Magic Numbers

Magic Number findings refer to numeric literals that are expressed in the codebase in their raw format and should otherwise be specified as constant contract variables aiding in their legibility and maintainability.

Compiler Error

Compiler Error findings refer to an error in the structure of the code that renders it impossible to compile using the specified version of the project.

Disclaimer

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.