MythX

Analysis e025fb48-7761-49c9-aeb2-d6f88c93eb30 MythX

Started	
Finished	Mon Jul 03 2023 19:04:18 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Client Tool	Mythx-Cli-0.7.3
Main Source File	Contracts/RewardsProxy.Sol

DETECTED VULNERABILITIES

HIGH	MEDIUM	LOW
0	0	2

ISSUES

UNKNOWN SWC-101 Arithmetic operation "++" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

contracts/RewardsProxy.sol

Locations

```
function streamRewards(StreamRequest[] calldata streamRequests) external override {
uint256 numRequest = streamRequests.length;
for (uint256 i = 0; i < numRequest; ++i) {
streamReward(streamRequests[i]);
}
```

UNKNOWN SWC-101 Arithmetic operation "+" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol

Locations

```
function safeIncreaseAllowance(IERC20 token, address spender, uint256 value) internal {
uint256 oldAllowance = token.allowance(address(this), spender);
_callOptionalReturn(token, abi.encodeWithSelector(token.approve.selector, spender, oldAllowance + value));
}
```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol`

Locations

```
uint256 oldAllowance = token.allowance(address(this), spender);
require(oldAllowance >= value, "SafeERC20: decreased allowance below zero");
_callOptionalReturn(token, abi.encodeWithSelector(token.approve.selector, spender, oldAllowance - value));
}
}
```

UNKNOWN

SWC-101

Arithmetic operation "+" discovered

This plugin produces issues to support false positive discovery within MythX.

Source file

`node_modules/@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol`

Locations

```
token.permit(owner, spender, value, deadline, v, r, s);
uint256 nonceAfter = token.nonces(owner);
require(nonceAfter == nonceBefore + 1, "SafeERC20: permit did not succeed");
}
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.8.6"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source file

`contracts/RewardsProxy.sol`

Locations

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.6;

import "./interfaces/rewardsProxy/IRewardsProxy.sol";
```

LOW

SWC-108

State variable visibility is not set.

It is best practice to set the visibility of state variables explicitly. The default visibility for "oracleRegistry" is internal. Other possible visibility settings are public and private.

Source file

`contracts/RewardsProxy.sol`

Locations

```
*/
contract RewardsProxy is IRewardsProxy {
IOracleRegistry immutable oracleRegistry;

constructor(address _oracleRegistry) {
```

UNKNOWN

SWC-110

Out of bounds array access

The index access expression can cause an exception in case of use of invalid array index value.

Source file

`contracts/RewardsProxy.sol`

Locations

```
uint256 numRequest = streamRequests.length;
for (uint256 i = 0; i < numRequest; ++i) {
streamReward(streamRequests[i]);
}
emit RewardsStreamed(streamRequests);
```