MythX

Analysis f638c6b2-fab8-4230-af1d-4f817706cc1c MythX

Started	Sat Dec 04 2021 13:04:06 GMT+0000 (Coordinated Universal Time)
Finished	Sat Dec 04 2021 13:04:15 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Client Tool	Remythx
Main Source File	Depositor.Sol

DETECTED VULNERABILITIES

HIGH	MEDIUM	LOW
0	0	1

ISSUES

UNKNOWN SWC-101 Arithmetic operation "+" discovered

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

Source file

Depositor.sol

Locations

```
402 uint256 value
403 ) internal {
404 uint256 newAllowance = token.allowance(address(this), spender) + value;
405 _callOptionalReturn(token, abi.encodeWithSelector(token.approve.selector, spender, newAllowance));
406 }
```

UNKNOWN SWC-101 Arithmetic operation "-" discovered

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

Source file

Depositor.sol

Locations

```
414 uint256 oldAllowance = token.allowance(address(this), spender);
415 require(oldAllowance >= value, "SafeERC20: decreased allowance below zero");
416 uint256 newAllowance = oldAllowance - value;
417 _callOptionalReturn(token, abi.encodeWithSelector(token.approve.selector, spender, newAllowance));
418 }
```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

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

Source file

Depositor.sol

Locations

```

uint index = map.indexOf[key];
uint lastIndex = map.keys.length - 1;
address lastKey = map.keys[lastIndex];
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

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

Source file

Depositor.sol

Locations

```
uint256 public totalPending;
uint256 private undestributedReward;
uint256 public minAmountToStake = 3 * 10 ** 6 * 10 ** 18;

bool public depositsEnabled = true;
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

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

Source file

Depositor.sol

Locations

```
uint256 public totalPending;
uint256 private undestributedReward;
uint256 public minAmountToStake = 3 * 10 ** 6 * 10 ** 18;

bool public depositsEnabled = true;
```

UNKNOWN

SWC-101

Arithmetic operation "**" discovered

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

Source file

Depositor.sol

Locations

```
uint256 public totalPending;
uint256 private undestributedReward;
uint256 public minAmountToStake = 3 * 10 ** 6 * 10 ** 18;

bool public depositsEnabled = true;
```

UNKNOWN

SWC-101

Arithmetic operation "**" discovered

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

Source file

Depositor.sol

Locations

```
uint256 public totalPending;
uint256 private undestributedReward;
uint256 public minAmountToStake = 3 * 10 ** 6 * 10 ** 18;

bool public depositsEnabled = true;
```

UNKNOWN

SWC-101

Arithmetic operation "+=" discovered

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

Source file

Depositor.sol

Locations

```
require(depositsEnabled);
token.safeTransferFrom(msg.sender, address(this), _amount);
totalPending += _amount;

if (totalPending >= minAmountToStake || totalStaked >= minAmountToStake) {
```

UNKNOWN

Arithmetic operation "+=" discovered

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

SWC-101

Source file

Depositor.sol

Locations

```
if (totalPending >= minAmountToStake || totalStaked >= minAmountToStake) {
mainContract.stake(totalPending);
totalStaked += totalPending;
totalPending = 0;
emit StakeSuccessfull(totalPending);
```

UNKNOWN

Arithmetic operation "+" discovered

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

SWC-101

Source file

Depositor.sol

Locations

```
if (investors.inserted[msg.sender]) {
uint256 oldAmount = investors.get(msg.sender);
investors.set(msg.sender, _amount + oldAmount);
} else {
investors.set(msg.sender, _amount);
```

UNKNOWN

Arithmetic operation "-" discovered

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

SWC-101

Source file

Depositor.sol

Locations

```
mainContract.unstake(false);
balanceAfter = token.balanceOf(address(this));
totalPending = balanceAfter - balanceBefore - amount;
totalStaked = 0;
} else {
```

UNKNOWN

SWC-101

Arithmetic operation "-" discovered

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

Source file

Depositor.sol

Locations

```
mainContract.unstake(false);
balanceAfter = token.balanceOf(address(this));
totalPending = balanceAfter - balanceBefore - amount;
totalStaked = 0;
} else {
```

UNKNOWN

SWC-101

Arithmetic operation "-=" discovered

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

Source file

Depositor.sol

Locations

```
totalStaked = 0;
} else {
totalPending -= amount;
}
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

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

Source file

Depositor.sol

Locations

```
_stake();

uint256 fee = amount * withdrawFeePercent / denominator;
amount -= fee;
// stakers[msg.sender] = 0;
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

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

Source file

Depositor.sol

Locations

```
_stake();

uint256 fee = amount * withdrawFeePercent / denominator;
amount -= fee;
// stakers[msg.sender] = 0;
```

UNKNOWN

SWC-101

Arithmetic operation "-=" discovered

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

Source file

Depositor.sol

Locations

```

uint256 fee = amount * withdrawFeePercent / denominator;
amount -= fee;
// stakers[msg.sender] = 0;
token.safeTransfer(msg.sender, amount);
```

UNKNOWN

SWC-101

Arithmetic operation "+=" discovered

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

Source file

Depositor.sol

Locations

```
if (totalPending >= minAmountToStake) {
mainContract.stake(totalPending);
totalStaked += totalPending;
totalPending = 0;
}
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

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

Source file

Depositor.sol

Locations

```
} else {
uint256 totalReward = mainContract.claimableReward();
uint256 stakeShare = investors.get(_address) * 10**18 / totalStaked;
// uint256 stakeShare = stakers[_address] * 10**18 / totalStaked;
return totalReward * stakeShare / 10**18;
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

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

Source file

Depositor.sol

Locations

```
} else {
uint256 totalReward = mainContract.claimableReward();
uint256 stakeShare = investors.get(_address) * 10**18 / totalStaked;
// uint256 stakeShare = stakers[_address] * 10**18 / totalStaked;
return totalReward * stakeShare / 10**18;
```

UNKNOWN

SWC-101

Arithmetic operation "**" discovered

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

Source file

Depositor.sol

Locations

```
} else {
uint256 totalReward = mainContract.claimableReward();
uint256 stakeShare = investors.get(_address) * 10**18 / totalStaked;
// uint256 stakeShare = stakers[_address] * 10**18 / totalStaked;
return totalReward * stakeShare / 10**18;
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

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

Source file

`Depositor.sol`

Locations

```
uint256 stakeShare = investors.get(_address) * 10**18 / totalStaked;
// uint256 stakeShare = stakers[_address] * 10**18 / totalStaked;
return totalReward * stakeShare / 10**18;
}
}
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

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

Source file

`Depositor.sol`

Locations

```
uint256 stakeShare = investors.get(_address) * 10**18 / totalStaked;
// uint256 stakeShare = stakers[_address] * 10**18 / totalStaked;
return totalReward * stakeShare / 10**18;
}
}
```

UNKNOWN

SWC-101

Arithmetic operation "**" discovered

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

Source file

`Depositor.sol`

Locations

```
uint256 stakeShare = investors.get(_address) * 10**18 / totalStaked;
// uint256 stakeShare = stakers[_address] * 10**18 / totalStaked;
return totalReward * stakeShare / 10**18;
}
}
```

UNKNOWN

SWC-101

Arithmetic operation "/" discovered

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

Source file

`Depositor.sol`

Locations

```

function _rewardAmount(address _address, uint256 _totalReward) internal view returns (uint256) {
uint256 stakeShare = investors.get(_address) * 10**18 / totalStaked;
// uint256 stakeShare = stakers[_address] * 10**18 / totalStaked;
return _totalReward * stakeShare / 10**18;
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

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

Source file

`Depositor.sol`

Locations

```

function _rewardAmount(address _address, uint256 _totalReward) internal view returns (uint256) {
uint256 stakeShare = investors.get(_address) * 10**18 / totalStaked;
// uint256 stakeShare = stakers[_address] * 10**18 / totalStaked;
return _totalReward * stakeShare / 10**18;
```

UNKNOWN

SWC-101

Arithmetic operation "**" discovered

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

Source file

`Depositor.sol`

Locations

```

function _rewardAmount(address _address, uint256 _totalReward) internal view returns (uint256) {
uint256 stakeShare = investors.get(_address) * 10**18 / totalStaked;
// uint256 stakeShare = stakers[_address] * 10**18 / totalStaked;
return _totalReward * stakeShare / 10**18;
```

UNKNOWN

Arithmetic operation "/" discovered

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

SWC-101

Source file

Depositor.sol

Locations

```
uint256 stakeShare = investors.get(_address) * 10**18 / totalStaked;
// uint256 stakeShare = stakers[_address] * 10**18 / totalStaked;
return _totalReward * stakeShare / 10**18;
}
```

UNKNOWN

Arithmetic operation "*" discovered

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

SWC-101

Source file

Depositor.sol

Locations

```
uint256 stakeShare = investors.get(_address) * 10**18 / totalStaked;
// uint256 stakeShare = stakers[_address] * 10**18 / totalStaked;
return _totalReward * stakeShare / 10**18;
}
```

UNKNOWN

Arithmetic operation "**" discovered

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

SWC-101

Source file

Depositor.sol

Locations

```
uint256 stakeShare = investors.get(_address) * 10**18 / totalStaked;
// uint256 stakeShare = stakers[_address] * 10**18 / totalStaked;
return _totalReward * stakeShare / 10**18;
}
```

UNKNOWN

Arithmetic operation "/" discovered

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

SWC-101

Source file

Depositor.sol

Locations

```
require(totalStaked >= minAmountToStake);
(uint256 claimed, ) = mainContract.claimReward();
uint256 fee = claimed * harvestFeePercent / denominator;
claimed -= fee;
address investor;
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

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

Source file

Depositor.sol

Locations

```
require(totalStaked >= minAmountToStake);
(uint256 claimed, ) = mainContract.claimReward();
uint256 fee = claimed * harvestFeePercent / denominator;
claimed -= fee;
address investor;
```

UNKNOWN

SWC-101

Arithmetic operation "-=" discovered

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

Source file

Depositor.sol

Locations

```
(uint256 claimed, ) = mainContract.claimReward();
uint256 fee = claimed * harvestFeePercent / denominator;
claimed -= fee;
address investor;
for (uint256 i = 0; i < investors.size(); i++){
```

UNKNOWN

SWC-101

Arithmetic operation "++" discovered

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

Source file

Depositor.sol

Locations

```
claimed -= fee;
address investor;
for (uint256 i = 0; i < investors.size(); i++){
investor = investors.getKeyAtIndex(i);
uint256 userReward = _rewardAmount(investor, claimed);
```

UNKNOWN

SWC-101

Compiler-rewritable "<uint> - 1" discovered

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

Source file

`Depositor.sol`

Locations

```

uint index = map.indexOf[key];
uint lastIndex = map.keys.length - 1;
address lastKey = map.keys[lastIndex];
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.8.9"". 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

`Depositor.sol`

Locations

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.9;

interface IERC20 {
```

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

`Depositor.sol`

Locations

```

function getKeyAtIndex(Map storage map, uint index) internal view returns (address) {
return map.keys[index];
}
```

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

`Depositor.sol`

Locations

```
uint index = map.indexOf[key];
uint lastIndex = map.keys.length - 1;
address lastKey = map.keys[lastIndex];

map.indexOf[lastKey] = index;
```

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

`Depositor.sol`

Locations

```
delete map.indexOf[key];

map.keys[index] = lastKey;
map.keys.pop();
}
```