MythX

REPORT 6195B971E74D4100186C0CE1

Created	Thu Nov 18 2021 02:24:49 GMT+0000 (Coordinated Universal Time)
Number of analyses	1
User	5f50e9c4f992e6001848d9db

REPORT SUMMARY

Analyses ID	Main source file	Detected vulnerabilities
ff9a9ee4-8367-4e92-84e4-3c3849d5f6ab	treasury.sol	5

Analysis `ff9a9ee4-8367-4e92-84e4-3c3849d5f6ab` MythX

Started	Thu Nov 18 2021 02:24:52 GMT+0000 (Coordinated Universal Time)
Finished	Thu Nov 18 2021 03:10:12 GMT+0000 (Coordinated Universal Time)
Mode	Deep
Client Tool	Remythx
Main Source File	`Treasury.Sol`

DETECTED VULNERABILITIES

HIGH	MEDIUM	LOW
0	0	5

ISSUES

UNKNOWN Arithmetic operation "+" discovered

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

SWC-101

Source file
`treasury.sol`
Locations

```
499 */
500 function add(uint256 a, uint256 b) internal pure returns (uint256) {
501 uint256 c = a + b;
502 require(c >= a, "SafeMath: addition overflow");
```

UNKNOWN Arithmetic operation "-" discovered

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

SWC-101

Source file
`treasury.sol`
Locations

```
531 function sub(uint256 a, uint256 b, string memory errorMessage) internal pure returns (uint256) {
532 require(b <= a, errorMessage);
533 uint256 c = a - b;
534
535 return c;
```

UNKNOWN Arithmetic operation "*" discovered

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

SWC-101

Source file

treasury.sol

Locations

```
554 }
555
556 uint256 c = a * b;
557 require(c / a == b, "SafeMath: multiplication overflow");
```

UNKNOWN Arithmetic operation "/" discovered

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

SWC-101

Source file

treasury.sol

Locations

```
555
556 uint256 c = a * b;
557 require(c / a == b, "SafeMath: multiplication overflow");
558
559 return c;
```

UNKNOWN Arithmetic operation "/" discovered

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

SWC-101

Source file

treasury.sol

Locations

```
590 function div(uint256 a, uint256 b, string memory errorMessage) internal pure returns (uint256) {
591 require(b > 0, errorMessage);
592 uint256 c = a / b;
593 // assert(a == b * c + a % b); // There is no case in which this doesn't hold
```

UNKNOWN Arithmetic operation "%" discovered

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

SWC-101

Source file

treasury.sol

Locations

```
626 function mod(uint256 a, uint256 b, string memory errorMessage) internal pure returns (uint256) {
627 require(b != 0, errorMessage);
628 return a % b;
629 }
630 }
```

UNKNOWN Arithmetic operation "*" discovered

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

SWC-101

Source file

treasury.sol

Locations

```
655 function getCollateralSupply() public view returns (uint){
656 if(msg.sender == address(buck) && collateral.decimals() < 18 || msg.sender == address(buckPool) && collateral.decimals() < 18){
657 return collateral.balanceOf(address(this)) * 10 ** (uint(18) - collateral.decimals());
658 } else{
659 return collateral.balanceOf(address(this));
```

UNKNOWN Arithmetic operation "**" discovered

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

SWC-101

Source file

treasury.sol

Locations

```
655 function getCollateralSupply() public view returns (uint){
656 if(msg.sender == address(buck) && collateral.decimals() < 18 || msg.sender == address(buckPool) && collateral.decimals() < 18){
657 return collateral.balanceOf(address(this)) * 10 ** (uint(18) - collateral.decimals());
658 } else{
659 return collateral.balanceOf(address(this));
```

UNKNOWN Arithmetic operation "-" discovered

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

SWC-101

Source file

treasury.sol

Locations

```
655 function getCollateralSupply() public view returns (uint){
656 if(msg.sender == address(buck) && collateral.decimals() < 18 || msg.sender == address(buckPool) && collateral.decimals() < 18){
657 return collateral.balanceOf(address(this)) * 10 ** (uint(18) - collateral.decimals());
658 } else{
659 return collateral.balanceOf(address(this));
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""=0.6.0<0.8.0"". 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

`treasury.sol`

Locations

```
5   // SPDX-License-Identifier: MIT
6
7   pragma solidity >=0.6.0 <0.8.0;
8
9   /*
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""=0.6.0<0.8.0"". 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

`treasury.sol`

Locations

```
29
30
31  pragma solidity >=0.6.0 <0.8.0;
32
33  /**
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""=0.6.0<0.8.0"". 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

`treasury.sol`

Locations

```
105 }
106
107 pragma solidity >=0.6.0 <0.8.0;
108
109 /**
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.0<0.8.0"". 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

`treasury.sol`

Locations

```
406 }
407
408 pragma solidity ^0.6.0 <0.8.0;
409
410 /**
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.0<0.8.0"". 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

`treasury.sol`

Locations

```
472 }
473
474 pragma solidity ^0.6.0 <0.8.0;
475
476 /**
```