MythX

REPORT 652D4C40BFE98400197ADC0B

Created	Mon Oct 16 2023 14:44:16 GMT+0000 (Coordinated Universal Time)
Number of analyses	2
User	6515ed04f4bf58cffb592f02

REPORT SUMMARY

Analyses ID	Main source file	Detected vulnerabilities
e99e8682-f2b3-49fb-9a40-2fc9ff39999c	src/contracts/core/LotteryContract.sol	1
31fb67ea-6794-4f81-8ad1-84399c299cac	permissions/PauserRegistry.sol	0

Analysis e99e8682-f2b3-49fb-9a40-2fc9ff39999c MythX

Started

Finished Mon Oct 16 2023 14:44:23 GMT+0000 (Coordinated Universal Time)

Mode Standard

Client Tool Mythx-Cli-0.7.3

Main Source File Src/Contracts/Core/LotteryContract.Sol

DETECTED VULNERABILITIES

HIGH	MEDIUM	LOW
0	0	1

ISSUES

UNKNOWN Arithmetic operation "-" discovered

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

SWC-101

Source file

lib/openzeppelin-contracts/contracts/utils/structs/EnumerableSet.sol

Locations

```
91  // This modifies the order of the array, as noted in {at}.
92
93  uint256 valueIndex = position - 1;
94  uint256 lastIndex = set._values.length - 1;
```

UNKNOWN Arithmetic operation "-" discovered

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

SWC-101

Source file

lib/openzeppelin-contracts/contracts/utils/structs/EnumerableSet.sol

Locations

```
92
93  uint256 valueIndex = position - 1;
94  uint256 lastIndex = set._values.length - 1;
95
96  if (valueIndex != lastIndex) {
```

UNKNOWN Compiler-rewritable "<uint> - 1" discovered

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

SWC-101

Source file

lib/openzeppelin-contracts/contracts/utils/structs/EnumerableSet.sol

Locations

```
91  // This modifies the order of the array, as noted in {at}.
92
93  uint256 valueIndex = position - 1;
94  uint256 lastIndex = set._values.length - 1;
```

UNKNOWN Compiler-rewritable "<uint> - 1" discovered

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

SWC-101

Source file

lib/openzeppelin-contracts/contracts/utils/structs/EnumerableSet.sol

Locations

```
92
93  uint256 valueIndex = position - 1;
94  uint256 lastIndex = set._values.length - 1;
95
96  if (valueIndex != lastIndex) {
```

UNKNOWN Arithmetic operation "-" discovered

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

SWC-101

Source file

src/contracts/core/LotteryContract.sol

Locations

```
115
116 modifier onlyTimeValid(uint256 timeDuration) {
117 require(block.timestamp - lastBlockTimestamp < timeDuration, "Invalid call, your time period has expired");
118 _;
119 }
```

UNKNOWN Arithmetic operation "+" discovered

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

SWC-101

Source file

src/contracts/core/LotteryContract.sol

Locations

```
137 VRFConsumerBaseV2(_chainlinkVrfCoordinator)
138 {
139 require(_lotteryDuration > _prizeExpiry + _expiryGap, "LotteryContract.constructor: Insufficient gap between expiry and duration");
140 lotteryDeposit = _lotteryDeposit;
141 lotteryDuration = _lotteryDuration;
```

UNKNOWN Arithmetic operation "+=" discovered

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

SWC-101

Source file

src/contracts/core/LotteryContract.sol

Locations

```
186 require(!lotteryUsers.contains(msg.sender), "LotteryContract.enterLottery: Not allow more than 1 deposit in given session");
187 require(msg.value == lotteryDeposit, "LotteryContract.enterLottery: Incorrect amount deposited");
188 currentPrizePot += msg.value;
189 lotteryUsers.add(msg.sender);
190 emit LotteryDeposit(msg.sender, msg.value);
```

UNKNOWN Arithmetic operation "-=" discovered

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

SWC-101

Source file

src/contracts/core/LotteryContract.sol

Locations

```
223 uint256 prizePayout = lotteryWinnings;
224 lotteryWinnings = 0; // Resetting snapshot
225 currentPrizePot -= prizePayout;
226
227 // 2300 gas stipend
```

UNKNOWN Arithmetic operation "-" discovered

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

SWC-101

Source file

src/contracts/core/LotteryContract.sol

Locations

```
242 require(withdrawalAmount > 0, "LotteryContract.withdraw: Cannot withdraw 0 Ether");
243 //must only be able to withdraw excess of total sum of current user deposits, i.e. require account.balance > sum(user deposits)
244 uint256 protocolFees = address(this).balance - currentPrizePot;
245 require(withdrawalAmount <= protocolFees, "LotteryContract.withdraw: Insufficient protocol funds for withdrawal amount");
246 (bool success, ) = msg.sender.call{gas: 2300, value: withdrawalAmount}("");
```

UNKNOWN Arithmetic operation "-" discovered

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

SWC-101

Source file

src/contracts/core/LotteryContract.sol

Locations

```
307 */
308 function checkUpkeep(bytes memory /* checkData */) public view returns (bool upkeepNeeded, bytes memory /* performData */) {
309 if((lotteryUsers.length() == 0) || ((block.timestamp - lastBlockTimestamp) < lotteryDuration)) {
310 return (false, ""); // Return an empty bytes value for the unused parameter
311 } else {
```

UNKNOWN Arithmetic operation "%" discovered

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

SWC-101

Source file

src/contracts/core/LotteryContract.sol

Locations

```
324 function fulfillRandomWords(uint256 /* _requestId */, uint256[] memory _randomWords) internal override whenNotPaused {
325 // Bound the winning index to the number of entrants in current round
326 uint256 winnerIndex = _randomWords[0] % lotteryUsers.length();
327 currentWinner = payable(lotteryUsers.at(winnerIndex));
```

UNKNOWN Arithmetic operation "/" discovered

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

SWC-101

Source file

src/contracts/core/LotteryContract.sol

Locations

```
328
329 // 50bps protocol fee subtracted from winning pot, regardless if winner claims or not
330 currentPrizePot = (currentPrizePot * 995) / 1000;
331
332 // Snapshots session ending pot size and time
```

UNKNOWN Arithmetic operation "*" discovered

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

SWC-101

Source file

src/contracts/core/LotteryContract.sol

Locations

```
328
329 // 50bps protocol fee subtracted from winning pot, regardless if winner claims or not
330 currentPrizePot = (currentPrizePot * 995) / 1000;
331
332 // Snapshots session ending pot size and time
```

UNKNOWN Arithmetic operation "++" discovered

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

SWC-101

Source file

src/contracts/core/LotteryContract.sol

Locations

```
343 function clearSet() private {
344 uint256 length = lotteryUsers.length();
345 for (uint256 i = 0; i < length; i++) {
346 // Always remove the first element until the set is empty
347 lotteryUsers.remove(lotteryUsers.at(0));
```

LOW

Use of "tx.origin" as a part of authorization control.

Using "tx.origin" as a security control can lead to authorization bypass vulnerabilities. Consider using "msg.sender" unless you really know what you are doing.

SWC-115

Source file

src/contracts/core/LotteryContract.sol

Locations

```
163 function initialize(IPauserRegistry _pauserRegistry) external initializer {
164 pauserRegistry = _pauserRegistry; // Set pauser registry
165 __Ownable_init(tx.origin); // Initialize ownable with tx.origin
166 __Pausable_init(); // Initialize pausable
167 __ReentrancyGuard_init(); // Initialize reentrancyguard
```

UNKNOWN

Out of bounds array access

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

SWC-110

Source file

lib/openzeppelin-contracts/contracts/utils/structs/EnumerableSet.sol

Locations

```
95
96 if (valueIndex != lastIndex) {
97 bytes32 lastValue = set._values[lastIndex];
98
99 // Move the lastValue to the index where the value to delete is
```

UNKNOWN

Out of bounds array access

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

SWC-110

Source file

lib/openzeppelin-contracts/contracts/utils/structs/EnumerableSet.sol

Locations

```
98
99 // Move the lastValue to the index where the value to delete is
100 set._values[valueIndex] = lastValue;
101 // Update the tracked position of the lastValue (that was just moved)
102 set._positions[lastValue] = position;
```

UNKNOWN

Out of bounds array access

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

SWC-110

Source file

lib/openzeppelin-contracts/contracts/utils/structs/EnumerableSet.sol

Locations

```
140 */
141 function _at(Set storage set, uint256 index) private view returns (bytes32) {
142 return set._values[index];
143 }
```

UNKNOWN

Out of bounds array access

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

SWC-110

Source file

src/contracts/core/LotteryContract.sol

Locations

```
324 function fulfillRandomWords(uint256 /* _requestId */, uint256[] memory _randomWords) internal override whenNotPaused {
325 // Bound the winning index to the number of entrants in current round
326 uint256 winnerIndex = _randomWords[0] % lotteryUsers.length();
327 currentWinner = payable(lotteryUsers.at(winnerIndex));
```

Analysis 31fb67ea-6794-4f81-8ad1-84399c299cac MythX

Started

Finished Mon Oct 16 2023 14:44:24 GMT+0000 (Coordinated Universal Time)

Mode Standard

Client Tool Mythx-Cli-0.7.3

Main Source File Permissions/PauserRegistry.Sol

DETECTED VULNERABILITIES

HIGH	MEDIUM	LOW
0	0	0

ISSUES

UNKNOWN Arithmetic operation "++" discovered

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

SWC-101

Source file

permissions/PauserRegistry.sol

Locations

```
30
31 // Set initial pausers
32 for (uint256 i = 0; i < initialPausers.length; i++) {
33 _updatePauser(initialPausers[i], true);
34 }
```

UNKNOWN Arithmetic operation "++" discovered

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

SWC-101

Source file

permissions/PauserRegistry.sol

Locations

```
35
36 // Set initial unpausers
37 for (uint256 i = 0; i < initialUnpausers.length; i++) {
38 _updateUnpauser(initialUnpausers[i], true);
39 }
```

UNKNOWN **Out of bounds array access**

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

SWC-110

Source file

permissions/PauserRegistry.sol

Locations

```
31 // Set initial pausers
32 for (uint256 i = 0; i < initialPausers.length; i++) {
33 _updatePauser(initialPausers[i], true);
34 }
```

UNKNOWN **Out of bounds array access**

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

SWC-110

Source file

permissions/PauserRegistry.sol

Locations

```
36 // Set initial unpausers
37 for (uint256 i = 0; i < initialUnpausers.length; i++) {
38 _updateUnpauser(initialUnpausers[i], true);
39 }
40 }
```