MythX

Analysis 881b8ae9-25f4-4ca3-80fd-37e57ee9720c MythX

Started	Thu Nov 03 2022 15:58:00 GMT+0000 (Coordinated Universal Time)
Finished	Thu Nov 03 2022 16:02:02 GMT+0000 (Coordinated Universal Time)
Mode	Quick
Client Tool	Remythx
Main Source File	LuckyELON.Sol

DETECTED VULNERABILITIES

HIGH	MEDIUM	LOW
0	0	19

ISSUES

UNKNOWN Arithmetic operation "+" discovered

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

SWC-101

Source file
LuckyELON.sol
Locations

```
19 */
20 function add(uint256 a, uint256 b) internal pure returns (uint256) {
21 uint256 c = a + b;
22 require(c >= a, "SafeMath: addition overflow");
23
```

UNKNOWN Arithmetic operation "-" discovered

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

SWC-101

Source file
LuckyELON.sol
Locations

```
51 function sub(uint256 a, uint256 b, string memory errorMessage) internal pure returns (uint256) {
52 require(b <= a, errorMessage);
53 uint256 c = a - b;
54
55 return c;
```

UNKNOWN

Arithmetic operation "*" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
74  }
75
76  uint256 c = a * b;
77  require(c / a == b, "SafeMath: multiplication overflow");
78
```

UNKNOWN

Arithmetic operation "/" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
75
76  uint256 c = a * b;
77  require(c / a == b, "SafeMath: multiplication overflow");
78
79  return c;
```

UNKNOWN

Arithmetic operation "/" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
110 function div(uint256 a, uint256 b, string memory errorMessage) internal pure returns (uint256) {
111 require(b > 0, errorMessage);
112 uint256 c = a / b;
113 // assert(a == b * c + a % b); // There is no case in which this doesn't hold
114
```

UNKNOWN

Arithmetic operation "%" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
146 function mod(uint256 a, uint256 b, string memory errorMessage) internal pure returns (uint256) {
147 require(b != 0, errorMessage);
148 return a % b;
149 }
150 }
```

UNKNOWN

Arithmetic operation "*" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
160 */
161 function mul(int256 a, int256 b) internal pure returns (int256) {
162 int256 c = a * b;
163
164 // Detect overflow when multiplying MIN_INT256 with -1
```

UNKNOWN

Arithmetic operation "/" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
164 // Detect overflow when multiplying MIN_INT256 with -1
165 require(c != MIN_INT256 || (a & MIN_INT256) != (b & MIN_INT256));
166 require((b == 0) || (c / b == a));
167 return c;
168 }
```

UNKNOWN

Arithmetic operation "/" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
176
177 // Solidity already throws when dividing by 0.
178 return a / b;
179 }
180
```

UNKNOWN

Arithmetic operation "-" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
183 */
184 function sub(int256 a, int256 b) internal pure returns (int256) {
185 int256 c = a - b;
186 require((b >= 0 && c <= a) || (b < 0 && c > a));
187 return c;
```

UNKNOWN

Arithmetic operation "+" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
192 */
193 function add(int256 a, int256 b) internal pure returns (int256) {
194 int256 c = a + b;
195 require((b >= 0 && c >= a) || (b < 0 && c < a));
196 return c;
```

UNKNOWN Arithmetic operation "-" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
344
345 uint index = map.indexOf[key];
346 uint lastIndex = map.keys.length - 1;
347 address lastKey = map.keys[lastIndex];
348
```

UNKNOWN Arithmetic operation "**" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1005 // For more discussion about choosing the value of `magnitude`,
1006 // see https://github.com/ethereum/EIPs/issues/1726#issuecomment-472352728
1007 uint256 constant internal magnitude = 2**128;
1008
1009 uint256 internal magnifiedDividendPerShare;
```

UNKNOWN Arithmetic operation "/" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1052 if (msg.value > 0) {
1053 magnifiedDividendPerShare = magnifiedDividendPerShare.add(
1054 (msg.value).mul(magnitude) / totalSupply()
1055 );
1056 emit DividendsDistributed(msg.sender, msg.value);
```

UNKNOWN

Arithmetic operation "/" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1115 /// @return The amount of dividend in wei that `_owner` has earned in total.
1116 function accumulativeDividendOf(address _owner) public view override returns(uint256) {
1117 return magnifiedDividendPerShare.mul(balanceOf(_owner)).toInt256Safe()
1118 .add(magnifiedDividendCorrections[_owner]).toUint256Safe() / magnitude;
1119 }
1120
```

UNKNOWN

Arithmetic operation "*" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1185 uint256 public lotteryFireAmount = 100000000000000000;
1186
1187 uint256 public maxSellTransactionAmount = 1 * 10 ** 12 * (10**9);
1188 uint256 public swapTokensAtAmount = 1 * 10 ** 9 * (10**9);
1189 uint256 public maxHoldAmount = 1 * 10 ** 12 * (10 ** 9);
```

UNKNOWN

Arithmetic operation "*" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1185 uint256 public lotteryFireAmount = 100000000000000000;
1186
1187 uint256 public maxSellTransactionAmount = 1 * 10 ** 12 * (10**9);
1188 uint256 public swapTokensAtAmount = 1 * 10 ** 9 * (10**9);
1189 uint256 public maxHoldAmount = 1 * 10 ** 12 * (10 ** 9);
```

UNKNOWN Arithmetic operation "**" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1185 uint256 public lotteryFireAmount = 100000000000000000;
1186
1187 uint256 public maxSellTransactionAmount = 1 * 10 ** 12 * (10**9);
1188 uint256 public swapTokensAtAmount = 1 * 10 ** 9 * (10**9);
1189 uint256 public maxHoldAmount = 1 * 10 ** 12 * (10 ** 9);
```

UNKNOWN Arithmetic operation "**" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1185 uint256 public lotteryFireAmount = 100000000000000000;
1186
1187 uint256 public maxSellTransactionAmount = 1 * 10 ** 12 * (10**9);
1188 uint256 public swapTokensAtAmount = 1 * 10 ** 9 * (10**9);
1189 uint256 public maxHoldAmount = 1 * 10 ** 12 * (10 ** 9);
```

UNKNOWN Arithmetic operation "*" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1186
1187 uint256 public maxSellTransactionAmount = 1 * 10 ** 12 * (10**9);
1188 uint256 public swapTokensAtAmount = 1 * 10 ** 9 * (10**9);
1189 uint256 public maxHoldAmount = 1 * 10 ** 12 * (10 ** 9);
1190
```

UNKNOWN Arithmetic operation "*" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1186
1187 uint256 public maxSellTransactionAmount = 1 * 10 ** 12 * (10**9);
1188 uint256 public swapTokensAtAmount = 1 * 10 ** 9 * (10**9);
1189 uint256 public maxHoldAmount = 1 * 10 ** 12 * (10 ** 9);
1190
```

UNKNOWN Arithmetic operation "**" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1186
1187 uint256 public maxSellTransactionAmount = 1 * 10 ** 12 * (10**9);
1188 uint256 public swapTokensAtAmount = 1 * 10 ** 9 * (10**9);
1189 uint256 public maxHoldAmount = 1 * 10 ** 12 * (10 ** 9);
1190
```

UNKNOWN Arithmetic operation "**" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1186
1187 uint256 public maxSellTransactionAmount = 1 * 10 ** 12 * (10**9);
1188 uint256 public swapTokensAtAmount = 1 * 10 ** 9 * (10**9);
1189 uint256 public maxHoldAmount = 1 * 10 ** 12 * (10 ** 9);
1190
```

UNKNOWN Arithmetic operation "*" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1187 uint256 public maxSellTransactionAmount = 1 * 10 ** 12 * (10**9);
1188 uint256 public swapTokensAtAmount = 1 * 10 ** 9 * (10**9);
1189 uint256 public maxHoldAmount = 1 * 10 ** 12 * (10 ** 9);
1190
1191 uint256 public BNBRewardsFee = 4;
```

UNKNOWN Arithmetic operation "*" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1187 uint256 public maxSellTransactionAmount = 1 * 10 ** 12 * (10**9);
1188 uint256 public swapTokensAtAmount = 1 * 10 ** 9 * (10**9);
1189 uint256 public maxHoldAmount = 1 * 10 ** 12 * (10 ** 9);
1190
1191 uint256 public BNBRewardsFee = 4;
```

UNKNOWN Arithmetic operation "**" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1187 uint256 public maxSellTransactionAmount = 1 * 10 ** 12 * (10**9);
1188 uint256 public swapTokensAtAmount = 1 * 10 ** 9 * (10**9);
1189 uint256 public maxHoldAmount = 1 * 10 ** 12 * (10 ** 9);
1190
1191 uint256 public BNBRewardsFee = 4;
```

UNKNOWN Arithmetic operation "**" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1187 uint256 public maxSellTransactionAmount = 1 * 10 ** 12 * (10**9);
1188 uint256 public swapTokensAtAmount = 1 * 10 ** 9 * (10**9);
1189 uint256 public maxHoldAmount = 1 * 10 ** 12 * (10 ** 9);
1190
1191 uint256 public BNBRewardsFee = 4;
```

UNKNOWN Arithmetic operation "*" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1195 uint256 public totalFees = 12;
1196
1197 uint256 public eligibleAmountForLottery = 100 * 10 ** 6 * 10 ** 9;
1198
1199 address [] public _listOfHolders;
```

UNKNOWN Arithmetic operation "*" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1195 uint256 public totalFees = 12;
1196
1197 uint256 public eligibleAmountForLottery = 100 * 10 ** 6 * 10 ** 9;
1198
1199 address [] public _listOfHolders;
```

UNKNOWN

SWC-101

Arithmetic operation "**" discovered

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

Source file

LuckyELON.sol

Locations

```
1195 uint256 public totalFees = 12;
1196
1197 uint256 public eligibleAmountForLottery = 100 * 10 ** 6 * 10 ** 9;
1198
1199 address [] public _listOfHolders;
```

UNKNOWN

SWC-101

Arithmetic operation "**" discovered

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

Source file

LuckyELON.sol

Locations

```
1195 uint256 public totalFees = 12;
1196
1197 uint256 public eligibleAmountForLottery = 100 * 10 ** 6 * 10 ** 9;
1198
1199 address [] public _listOfHolders;
```

UNKNOWN

SWC-101

Arithmetic operation "*" discovered

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

Source file

LuckyELON.sol

Locations

```
1263 and CANNOT be called ever again
1264 */
1265 _mint(owner(), 100 * 10 ** 12 * (10**9));
1266 }
1267
```

UNKNOWN

Arithmetic operation "*" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1263 and CANNOT be called ever again
1264 */
1265 _mint(owner(), 100 * 10 ** 12 * (10**9));
1266 }
1267
```

UNKNOWN

Arithmetic operation "**" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1263 and CANNOT be called ever again
1264 */
1265 _mint(owner(), 100 * 10 ** 12 * (10**9));
1266 }
1267
```

UNKNOWN

Arithmetic operation "**" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1263 and CANNOT be called ever again
1264 */
1265 _mint(owner(), 100 * 10 ** 12 * (10**9));
1266 }
1267
```

UNKNOWN

Arithmetic operation "*" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1279
1280 function updateLotteryFireAmount(uint256 amount) public onlyOwner {
1281 lotteryFireAmount = amount * 1 ** 18;
1282 }
1283
```

UNKNOWN

Arithmetic operation "**" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1279
1280 function updateLotteryFireAmount(uint256 amount) public onlyOwner {
1281 lotteryFireAmount = amount * 1 ** 18;
1282 }
1283
```

UNKNOWN

Arithmetic operation "*" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1283
1284 function updateLotteryEligibleAmount(uint256 amount) public onlyOwner {
1285 eligibleAmountForLottery = amount * 10 ** 9;
1286 }
1287
```

UNKNOWN Arithmetic operation "**" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1283
1284 function updateLotteryEligibleAmount(uint256 amount) public onlyOwner {
1285 eligibleAmountForLottery = amount * 10 ** 9;
1286 }
1287
```

UNKNOWN Arithmetic operation "*" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1287
1288 function updateMaxHoldAmount (uint256 amount) public onlyOwner {
1289 maxHoldAmount = amount * 10 ** 9;
1290 }
1291
```

UNKNOWN Arithmetic operation "**" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1287
1288 function updateMaxHoldAmount (uint256 amount) public onlyOwner {
1289 maxHoldAmount = amount * 10 ** 9;
1290 }
1291
```

UNKNOWN Arithmetic operation "*" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1291
1292 function updateMaxSellAmount (uint256 amount) public onlyOwner {
1293 maxSellTransactionAmount = amount * 10 ** 9;
1294 }
1295
```

UNKNOWN Arithmetic operation "**" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1291
1292 function updateMaxSellAmount (uint256 amount) public onlyOwner {
1293 maxSellTransactionAmount = amount * 10 ** 9;
1294 }
1295
```

UNKNOWN Arithmetic operation "+" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1299 liquidityFee = newliquidityfee;
1300 BNBRewardsFee = newBNBRewardsfee;
1301 totalFees = newmarketingfee + newlotteryfee + newliquidityfee + newBNBRewardsfee;
1302 }
1303
```

UNKNOWN Arithmetic operation "+" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1299 liquidityFee = newliquidityfee;
1300 BNBRewardsFee = newBNBRewardsfee;
1301 totalFees = newmarketingfee + newlotteryfee + newliquidityfee + newBNBRewardsfee;
1302 }
1303
```

UNKNOWN Arithmetic operation "+" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1299 liquidityFee = newliquidityfee;
1300 BNBRewardsFee = newBNBRewardsfee;
1301 totalFees = newmarketingfee + newlotteryfee + newliquidityfee + newBNBRewardsfee;
1302 }
1303
```

UNKNOWN Arithmetic operation "-" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1508
1509 function removeholder(address shareholder) internal {
1510 _listOfHolders[_holderIndexes[shareholder]] = _listOfHolders[_listOfHolders.length-1];
1511 _holderIndexes[_listOfHolders[_listOfHolders.length-1]] = _holderIndexes[shareholder];
1512 _listOfHolders.pop();
```

UNKNOWN Arithmetic operation "-" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1509 function removeholder(address shareholder) internal {
1510 _listOfHolders[_holderIndexes[shareholder]] = _listOfHolders[_listOfHolders.length-1];
1511 _holderIndexes[_listOfHolders[_listOfHolders.length-1]] = _holderIndexes[shareholder];
1512 _listOfHolders.pop();
1513 _bAddedHolderList[shareholder] = false;
```

UNKNOWN Arithmetic operation "+" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1518 keccak256(
1519 abi.encodePacked(
1520 block.timestamp + block.difficulty +
1521 ((uint256(keccak256(abi.encodePacked(block.coinbase)))) / (now)) +
1522 block.gaslimit +
1523 ((uint256(keccak256(abi.encodePacked(msg.sender)))) / (now)) +
1524 block.number +
1525 salty
1526 )
1527 )
```

UNKNOWN Arithmetic operation "+" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1518 keccak256(
1519 abi.encodePacked(
1520 block.timestamp + block.difficulty +
1521 ((uint256(keccak256(abi.encodePacked(block.coinbase)))) / (now)) +
1522 block.gaslimit +
1523 ((uint256(keccak256(abi.encodePacked(msg.sender)))) / (now)) +
1524 block.number +
1525 salty
1526 )
```

UNKNOWN

Arithmetic operation "+" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1518 keccak256(
1519 abi.encodePacked(
1520 block.timestamp + block.difficulty +
1521 ((uint256(keccak256(abi.encodePacked(block.coinbase)))) / (now)) +
1522 block.gaslimit +
1523 ((uint256(keccak256(abi.encodePacked(msg.sender)))) / (now)) +
1524 block.number +
1525 salty
```

UNKNOWN

Arithmetic operation "+" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1518 keccak256(
1519 abi.encodePacked(
1520 block.timestamp + block.difficulty +
1521 ((uint256(keccak256(abi.encodePacked(block.coinbase)))) / (now)) +
1522 block.gaslimit +
1523 ((uint256(keccak256(abi.encodePacked(msg.sender)))) / (now)) +
1524 block.number +
```

UNKNOWN

Arithmetic operation "+" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1518 keccak256(
1519 abi.encodePacked(
1520 block.timestamp + block.difficulty +
1521 ((uint256(keccak256(abi.encodePacked(block.coinbase)))) / (now)) +
1522 block.gaslimit +
1523 ((uint256(keccak256(abi.encodePacked(msg.sender)))) / (now)) +
```

UNKNOWN

Arithmetic operation "+" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1518 keccak256(
1519 abi.encodePacked(
1520 block.timestamp + block.difficulty +
1521 ((uint256(keccak256(abi.encodePacked(block.coinbase)))) / (now)) +
1522 block.gaslimit +
```

UNKNOWN

Arithmetic operation "/" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1519 abi.encodePacked(
1520 block.timestamp + block.difficulty +
1521 ((uint256(keccak256(abi.encodePacked(block.coinbase)))) / (now)) +
1522 block.gaslimit +
1523 ((uint256(keccak256(abi.encodePacked(msg.sender)))) / (now)) +
```

UNKNOWN

Arithmetic operation "/" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1521 ((uint256(keccak256(abi.encodePacked(block.coinbase)))) / (now)) +
1522 block.gaslimit +
1523 ((uint256(keccak256(abi.encodePacked(msg.sender)))) / (now)) +
1524 block.number +
1525 salty
```

UNKNOWN

Arithmetic operation "+" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1527 )
1528 );
1529 return seed.mod(to - from) + from;
1530 }
1531
```

UNKNOWN

Arithmetic operation "-" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1527 )
1528 );
1529 return seed.mod(to - from) + from;
1530 }
1531
```

UNKNOWN

Arithmetic operation "%" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1532 function _awardRandomEligibleHolder() private {
1533 if (_listOfHolders.length > 0){
1534 uint256 rndVal = random(100, 1000000, address(lotteryContract).balance) % _listOfHolders.length;
1535 lotteryContract.withdraw(_listOfHolders[rndVal]);
1536 }
```

UNKNOWN

Arithmetic operation "-" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1600 function swapAndSendDividends(uint256 tokens) private {
1601 swapTokensForEth(tokens);
1602 uint256 marketingBNB = address(this).balance.div(totalFees - liquidityFee).mul(marketingfee);
1603 uint256 lotteryBNB = address(this).balance.div(totalFees - liquidityFee).mul(lotteryFee);
1604 (bool success,) = address(devWallet).call{value: marketingBNB}("");
```

UNKNOWN

Arithmetic operation "-" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1601 swapTokensForEth(tokens);
1602 uint256 marketingBNB = address(this).balance.div(totalFees - liquidityFee).mul(marketingfee);
1603 uint256 lotteryBNB = address(this).balance.div(totalFees - liquidityFee).mul(lotteryFee);
1604 (bool success,) = address(devWallet).call{value: marketingBNB}("");
1605 (success,) = address(lotteryContract).call{value: lotteryBNB}("");
```

UNKNOWN

Arithmetic operation "*" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1635 constructor() public DividendPayingToken("_Dividend_Tracker", "_Dividend_Tracker") {
1636 claimWait = 3600;
1637 minimumTokenBalanceForDividends = 100000000 * (10**9); //must hold 100000000+ tokens
1638 }
1639
```

UNKNOWN

Arithmetic operation "**" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1635 constructor() public DividendPayingToken("_Dividend_Tracker", "_Dividend_Tracker") {
1636 claimWait = 3600;
1637 minimumTokenBalanceForDividends = 100000000 * (10**9); //must hold 100000000+ tokens
1638 }
1639
```

UNKNOWN

Arithmetic operation "++" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1777
1778 while(gasleft() > 100000 && iterations < numberOfTokenHolders) {
1779 _lastProcessedIndex++;
1780
1781 if(_lastProcessedIndex >= tokenHoldersMap.keys.length) {
```

UNKNOWN

Arithmetic operation "++" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1787 if(canAutoClaim(lastClaimTimes[account]) && withdrawableDividendOf(account) > 0) {
1788 if(processAccount(payable(account), true)) {
1789 claims++;
1790 }
1791 }
```

UNKNOWN Arithmetic operation "++" discovered

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1791 }
1792
1793 iterations++;
1794 }
1795
```

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

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

SWC-101

Source file

LuckyELON.sol

Locations

```
344
345 uint index = map.indexOf[key];
346 uint lastIndex = map.keys.length - 1;
347 address lastKey = map.keys[lastIndex];
348
```

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

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

SWC-101

Source file

LuckyELON.sol

Locations

```
1508
1509 function removeholder(address shareholder) internal {
1510 _listOfHolders[_holderIndexes[shareholder]] = _listOfHolders[_listOfHolders.length-1];
1511 _holderIndexes[_listOfHolders[_listOfHolders.length-1]] = _holderIndexes[shareholder];
1512 _listOfHolders.pop();
```

UNKNOWN

SWC-101

Compiler-rewritable "<uint> - 1" discovered

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

Source file

LuckyELON.sol

Locations

```
1509 function removeholder(address shareholder) internal {
1510 _listOfHolders[_holderIndexes[shareholder]] = _listOfHolders[_listOfHolders.length-1];
1511 _holderIndexes[_listOfHolders[_listOfHolders.length-1]] = _holderIndexes[shareholder];
1512 _listOfHolders.pop();
1513 _bAddedHolderList[shareholder] = false;
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.2"". 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

LuckyELON.sol

Locations

```
5 // SPDX-License-Identifier: MIT
6
7 pragma solidity ^0.6.2;
8
9 library SafeMath {
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.2"". 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

LuckyELON.sol

Locations

```
150 }
151
152 pragma solidity ^0.6.2;
153
154 library SafeMathInt {
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.2"". 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

LuckyELON.sol

Locations

```
212 }
213
214 pragma solidity ^0.6.2;
215
216 library SafeMathUint {
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.2"". 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

LuckyELON.sol

Locations

```
222 }
223
224 pragma solidity ^0.6.2;
225
226 abstract contract Context {
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.2"". 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

LuckyELON.sol

Locations

```
235 }
236
237 pragma solidity ^0.6.2;
238
239 contract Ownable is Context {
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.2"". 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

LuckyELON.sol

Locations

```
292 }
293
294 pragma solidity ^0.6.2;
295
296 library IterableMapping {
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.2"". 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

LuckyELON.sol

Locations

```
355 }
356
357 pragma solidity ^0.6.2;
358
359 interface IUniswapV2Pair {
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.2"". 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

LuckyELON.sol

Locations

```
408 }
409
410 pragma solidity ^0.6.2;
411
412 interface IUniswapV2Factory {
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.2"". 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

LuckyELON.sol

Locations

```
426 }
427
428 pragma solidity ^0.6.2;
429
430 interface IUniswapV2Router01 {
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.2"". 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

LuckyELON.sol

Locations

```
563 }
564
565 pragma solidity ^0.6.2;
566
567 interface IERC20 {
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.2"". 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

LuckyELON.sol

Locations

```
640 }
641
642 pragma solidity ^0.6.2;
643
644 interface IERC20Metadata is IERC20 {
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.2"". 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

LuckyELON.sol

Locations

```
659 }
660
661 pragma solidity ^0.6.2;
662
663 contract ERC20 is Context, IERC20, IERC20Metadata {
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.2"". 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

LuckyELON.sol

Locations

```
939 }
940
941 pragma solidity ^0.6.2;
942
943 interface DividendPayingTokenInterface {
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.2"". 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

LuckyELON.sol

Locations

```
975 }
976
977 pragma solidity ^0.6.2;
978
979 interface DividendPayingTokenOptionalInterface {
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.2"". 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

LuckyELON.sol

Locations

```
995 }
996
997 pragma solidity ^0.6.2;
998
999 contract DividendPayingToken is ERC20, DividendPayingTokenInterface, DividendPayingTokenOptionalInterface {
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.2"". 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

LuckyELON.sol

Locations

```
1167 }
1168
1169 pragma solidity ^0.6.2;
1170
1171 contract LUCKYELON is ERC20, Ownable {
```

LOW

SWC-115

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.

Source file

LuckyELON.sol

Locations

```
1383 require(gasleft() >= gas, "Out of gas, please increase gas limit and retry!");
1384 (uint256 iterations, uint256 claims, uint256 lastProcessedIndex) = dividendTracker.process{gas:gas}();
1385 emit ProcessedDividendTracker(iterations, claims, lastProcessedIndex, false, gas, tx.origin);
1386 }
1387
```

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

LuckyELON.sol

Locations

```
1482 require(gasleft() >= gas, "Out of gas, please increase gas limit and retry!");
1483 try dividendTracker.process{gas:gas}() returns (uint256 iterations, uint256 claims, uint256 lastProcessedIndex) {
1484 emit ProcessedDividendTracker(iterations, claims, lastProcessedIndex, true, gas, tx.origin);
1485 }
1486 catch {
```

UNKNOWN

Public state variable with array type causing reacheable exception by default.

The public state variable "_listOfHolders" in "LUCKYELON" contract has type "address[]" and can cause an exception in case of use of invalid array index value.

SWC-110

Source file

LuckyELON.sol

Locations

```
1197 uint256 public eligibleAmountForLottery = 100 * 10 ** 6 * 10 ** 9;
1198
1199 address [] public _listOfHolders;
1200 mapping (address => bool) public _bAddedHolderList;
1201 mapping (address => uint256) public _holderIndexes;
```

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

LuckyELON.sol

Locations

```
315
316 function getKeyAtIndex(Map storage map, uint index) public view returns (address) {
317 return map.keys[index];
318 }
319
```

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

LuckyELON.sol

Locations

```
345 uint index = map.indexOf[key];
346 uint lastIndex = map.keys.length - 1;
347 address lastKey = map.keys[lastIndex];
348
349 map.indexOf[lastKey] = index;
```

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

LuckyELON.sol

Locations

```
350 delete map.indexOf[key];
351
352 map.keys[index] = lastKey;
353 map.keys.pop();
354 }
```

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

LuckyELON.sol

Locations

```
1508
1509 function removeholder(address shareholder) internal {
1510 _listOfHolders[_holderIndexes[shareholder]] = _listOfHolders[_listOfHolders.length-1];
1511 _holderIndexes[_listOfHolders[_listOfHolders.length-1]] = _holderIndexes[shareholder];
1512 _listOfHolders.pop();
```

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

LuckyELON.sol

Locations

```
1508
1509 function removeholder(address shareholder) internal {
1510 _listOfHolders[_holderIndexes[shareholder]] = _listOfHolders[_listOfHolders.length-1];
1511 _holderIndexes[_listOfHolders[_listOfHolders.length-1]] = _holderIndexes[shareholder];
1512 _listOfHolders.pop();
```

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

LuckyELON.sol

Locations

```
1509 function removeholder(address shareholder) internal {
1510 _listOfHolders[_holderIndexes[shareholder]] = _listOfHolders[_listOfHolders.length-1];
1511 _holderIndexes[_listOfHolders[_listOfHolders.length-1]] = _holderIndexes[shareholder];
1512 _listOfHolders.pop();
1513 _bAddedHolderList[shareholder] = false;
```

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

LuckyELON.sol

Locations

```
1533 if (_listOfHolders.length > 0){
1534 uint256 rndVal = random(100, 1000000, address(lotteryContract).balance) % _listOfHolders.length;
1535 lotteryContract.withdraw(_listOfHolders[rndVal]);
1536 }
1537 }
```

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

LuckyELON.sol

Locations

```
1565 // generate the uniswap pair path of token -> weth
1566 address[] memory path = new address[](2);
1567 path[0] = address(this);
1568 path[1] = uniswapV2Router.WETH();
1569
```

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

LuckyELON.sol

Locations

```
1566 address[] memory path = new address[](2);
1567 path[0] = address(this);
1568 path[1] = uniswapV2Router.WETH();
1569
1570 _approve(address(this), address(uniswapV2Router), tokenAmount);
```

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

LuckyELON.sol

Locations

```
1783 }
1784
1785 address account = tokenHoldersMap.keys[_lastProcessedIndex];
1786
1787 if(canAutoClaim(lastClaimTimes[account]) && withdrawableDividendOf(account) > 0) {
```

LOW

SWC-120

Potential use of "block.number" as source of randonmness.

The environment variable "block.number" looks like it might be used as a source of randomness. Note that the values of variables like coinbase, gaslimit, block number and timestamp are predictable and can be manipulated by a malicious miner. Also keep in mind that attackers know hashes of earlier blocks. Don't use any of those environment variables as sources of randomness and be aware that use of these variables introduces a certain level of trust into miners.

Source file

LuckyELON.sol

Locations

```
1522 block.gaslimit +
1523 ((uint256(keccak256(abi.encodePacked(msg.sender)))) / (now)) +
1524 block.number +
1525 salty
1526 )
```