Super Moon Lotto

Smart Contract Audit Report

SYSFIXED

02 Dec 2022

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
Super Moon Lotto	SML	Binance Smart Chain

Addresses

Contract address	0x4d43e0b1eC8D829A4bB6ABaa8C2C41bF3c580A7F
Contract deployer address	0x0fbC2C9B1C65662Cdd969C5466414552833D50a5

Project Website

https://supermoonlotto.com/

Codebase

https://bscscan.com/address/0x4d43e0b1eC8D829A4bB6ABaa8C2C41bF3c580A7F#code

SUMMARY

New token mechanism is here! We aim to be the Biggest Lotto Game in the world ! Holders have FREE LOTTO TICKETS EVERY WEEK to win the weekly Jackpot.
1) 0.1 BTC added to the Jackpot monthly.
2) 4% of total transaction volume from the previous week will be added to weekly Jackpot (self generating)!
3) a pre-reserved number of tokens from Super Moon Lotto
***IF NO ONE WINS THE JACKPOT, we will take 5 balls away in the next draw, and we will keep taking 5 balls away until our holders hit the jackpot ***

Contract Summary

Documentation Quality

Super Moon Lotto provides a very good documentation with standard of solidity base code.

- The technical description is provided clearly and structured and also dont have any high risk issue.

Code Quality

The Overall quality of the basecode is standard.

- Standard solidity basecode and rules are already followed by Super Moon Lotto with the discovery of several low issues.

Test Coverage

Test coverage of the project is 100% (Through Codebase)

Audit Findings Summary

- SWC-100 SWC-108 | Explicitly define visibility for all state variables on lines 729, 734, 736, 737, 738 and 739.
- SWC-101 | It is recommended to use vetted safe math libraries for arithmetic operations consistently on lines 104, 136, 159, 160, 195, 231, 458, 702, 703, 741, 742, 768, 769, 899, 901, 949, 956, 958, 969, 1020, 1056, 1062, 1068, 1074, 1292, 1294, 1296, 1300, 1303, 1305, 1309, 1310, 1312, 1321, 1323, 1326, 1327, 1329, 1334, 1335, 1301, 1321, 1326 and 1327.
- SWC-103 | Pragma statements can be allowed to float when a contract is intended on lines 5.
- SWC-110 | It is recommended to use of revert(), assert(), and require() in Solidity, and the new REVERT opcode in the EVM on lines 727, 900, 901, 950, 959, 960, 1021, 1022, 1023, 1204, 1205, 1304, 1311, 1323, 1326, 1327, 1328, 1341, 1342, 1343, 1344 and 1345.

CONCLUSION

We have audited the Super Moon Lotto project released on December 2022 to discover issues and identify potential security vulnerabilities in Super Moon Lotto Project. This process is used to find technical issues and security loopholes which might be found in the smart contract.

The security audit report provides a satisfactory result with some low-risk issues.

The issues found in the code on Super Moon Lotto smart contract do not pose a considerable risk. The writing of the contract is close to the standard of writing contracts in general. The low-risk issues found are some arithmetic operation issues, a floating pragma is set, a state variable visibility is not set, a public state variable with array type causing reachable exception by default and out of bounds array access which the index access expression can cause an exception in case of the use of an invalid array index value.

AUDIT RESULT

Article	Category	Description	Result
Default Visibility	SWC-100 SWC-108	Functions and state variables visibility should be set explicitly. Visibility levels should be specified consciously.	ISSUE FOUND
Integer Overflow and Underflow	SWC-101	If unchecked math is used, all math operations should be safe from overflows and underflows.	ISSUE FOUND
Outdated Compiler Version	SWC-102	It is recommended to use a recent version of the Solidity compiler.	PASS
Floating Pragma	SWC-103	Contracts should be deployed with the same compiler version and flags that they have been tested thoroughly.	ISSUE FOUND
Unchecked Call Return Value	SWC-104	The return value of a message call should be checked.	PASS
SELFDESTRUCT Instruction	SWC-106	The contract should not be self-destructible while it has funds belonging to users.	PASS
Reentrancy	SWC-107	Check effect interaction pattern should be followed if the code performs recursive call.	PASS
Assert Violation	SWC-110	Properly functioning code should never reach a failing assert statement.	ISSUE FOUND
Deprecated Solidity Functions	SWC-111	Deprecated built-in functions should never be used.	PASS
Delegate call to Untrusted Callee	SWC-112	Delegate calls should only be allowed to trusted addresses.	PASS
DoS (Denial of Service)	SWC-113 SWC-128	Execution of the code should never be blocked by a specific contract state unless required.	PASS
Race Conditions	SWC-114	Race Conditions and Transactions Order Dependency should not be possible.	PASS

Authorization through tx.origin	SWC-115	tx.origin should not be used for authorization.	PASS
Block values as a proxy for time	SWC-116	Block numbers should not be used for time calculations.	PASS
Signature Unique ID	SWC-117 SWC-121 SWC-122	Signed messages should always have a unique id. A transaction hash should not be used as a unique id.	PASS
Shadowing State Variable	SWC-119	State variables should not be shadowed.	PASS
Weak Sources of Randomness	SWC-120	Random values should never be generated from Chain Attributes or be predictable.	PASS
Incorrect Inheritance Order	SWC-125	When inheriting multiple contracts, especially if they have identical functions, a developer should carefully specify inheritance in the correct order. The rule of thumb is to inherit contracts from more /general/ to more /specific/.	PASS

SMART CONTRACT ANALYSIS

Started	Thursday Dec 01 2022 22:28:27 GMT+0000 (Coordinated Universal Time)
Finished	Friday Dec 02 2022 23:37:41 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	SuperMoonLotto.sol

Detected Issues

ID	Title	Severity	Status
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "%" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged

SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged

SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "--" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED	low	acknowledged
SWC-101	COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED	low	acknowledged
SWC-101	COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED	low	acknowledged
SWC-101	COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED	low	acknowledged
SWC-101	COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED	low	acknowledged
SWC-101	COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-108	STATE VARIABLE VISIBILITY IS NOT SET.	low	acknowledged
SWC-108	STATE VARIABLE VISIBILITY IS NOT SET.	low	acknowledged
SWC-108	STATE VARIABLE VISIBILITY IS NOT SET.	low	acknowledged
SWC-108	STATE VARIABLE VISIBILITY IS NOT SET.	low	acknowledged
SWC-108	STATE VARIABLE VISIBILITY IS NOT SET.	low	acknowledged
SWC-108	STATE VARIABLE VISIBILITY IS NOT SET.	low	acknowledged
SWC-110	PUBLIC STATE VARIABLE WITH ARRAY TYPE CAUSING REACHABLE EXCEPTION BY DEFAULT.	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged

SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 104

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
103   function add(uint256 a, uint256 b) internal pure returns (uint256) {
104   uint256 c = a + b;
105   require(c >= a, "SafeMath: addition overflow");
106
107   return c;
108
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 136

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
135   require(b <= a, errorMessage);
136   uint256 c = a - b;
137
138   return c;
139   }
140
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 159

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
158
159   uint256 c = a * b;
160   require(c / a == b, "SafeMath: multiplication overflow");
161
162   return c;
163
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 160

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
159   uint256 c = a * b;
160   require(c / a == b, "SafeMath: multiplication overflow");
161
162   return c;
163   }
164
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 195

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
194   require(b > 0, errorMessage);
195   uint256 c = a / b;
196   // assert(a == b * c + a % b); // There is no case in which this doesn't hold
197
198   return c;
199
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 231

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
230   require(b != 0, errorMessage);
231   return a % b;
232   }
233   }
234
235
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 458

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
457   _owner = address(0);
458   _lockTime = block.timestamp + time;
459   emit OwnershipTransferred(_owner, address(0));
460   }
461
462
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 702

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
701   uint256 private constant MAX = ~uint256(0);
702   uint256 private _tTotal = 1000000 * 10**6 * 10**9; //CHANGE_OPTIONAL: total supply
of token. Recommended to keep unchanged
703   uint256 private _rTotal = (MAX - (MAX % _tTotal));
704   uint256 private _tFeeTotal;
705
706
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 703

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
702   uint256 private _tTotal = 1000000 * 10**6 * 10**9; //CHANGE_OPTIONAL: total supply
of token. Recommended to keep unchanged
703   uint256 private _rTotal = (MAX - (MAX % _tTotal));
704   uint256 private _tFeeTotal;
705
706   string private _name = "Super Moon Lotto";      //CHANGE_REQUIRED: name of token
707
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 741

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
740
741   uint256 public _maxTxAmount = 20000 * 10**6 * 10**9;
//CHANGE_OPTIONAL: max amount of tokens that can be transferred per transaction
742   uint256 private numTokensSellToAddToLiquidity = 20000 * 10**6 * 10**9;
//CHANGE_OPTIONAL: minimum number of tokens in contract to be sent to Pancakeswap pool
743   // uint256 public _jpPortion = 300 * 10**6 * 10**9;
744   event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);
745
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 742

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
741   uint256 public _maxTxAmount = 20000 * 10**6 * 10**9;
//CHANGE_OPTIONAL: max amount of tokens that can be transferred per transaction
742   uint256 private numTokensSellToAddToLiquidity = 20000 * 10**6 * 10**9;
//CHANGE_OPTIONAL: minimum number of tokens in contract to be sent to Pancakeswap pool
743   // uint256 public _jpPortion = 300 * 10**6 * 10**9;
744   event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);
745   event SwapAndLiquifyEnabledUpdated(bool enabled);
746
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 768

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
767   constructor (address dexAddress)  {
768   _rOwned[_msgSender()] = _rTotal.div(10**2).mul(90);
769   _rOwned[_moonJPAddress] = _rTotal.div(10**2).mul(10);
770
771   IUniswapV2Router02 _uniswapV2Router = IUniswapV2Router02(dexAddress);
772
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 769

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
768   _rOwned[_msgSender()] = _rTotal.div(10**2).mul(90);
769   _rOwned[_moonJPAddress] = _rTotal.div(10**2).mul(10);
770
771   IUniswapV2Router02 _uniswapV2Router = IUniswapV2Router02(dexAddress);
772   // Create a uniswap pair for this new token
773
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 899

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
898   require(_isExcluded[account], "Account is already excluded");
899   for (uint256 i = 0; i < _excluded.length; i++) {
900   if (_excluded[i] == account) {
901   _excluded[i] = _excluded[_excluded.length - 1];
902   _tOwned[account] = 0;
903
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 901

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
900   if (_excluded[i] == account) {
901   _excluded[i] = _excluded[_excluded.length - 1];
902   _tOwned[account] = 0;
903   _isExcluded[account] = false;
904   _excluded.pop();
905
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 949

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
948   uint256 eachPortion =
_rOwned[_moonWalletAddress].div(10).div(_numberOfSecondPrizeWinner);
949   for (uint256 i = 0; i < winningAddresses.length; i++) {
950   _rOwned[winningAddresses[i]] = _rOwned[winningAddresses[i]].add(eachPortion);
951   }
952   _rOwned[_moonWalletAddress] =
_rOwned[_moonWalletAddress].sub(_rOwned[_moonWalletAddress].div(10));
953
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 956

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
955   _numberOfFirstPrizeWinner = winningAddresses.length ;
956   uint256 _jpPortion = _rTotal.div(10**2).mul(10).mul(_jpRatio).div(10**3);
957   uint256 eachPortion = _rOwned[_moonWalletAddress].div(_numberOfFirstPrizeWinner);
958   for (uint256 i = 0; i < winningAddresses.length; i++) {
959   _rOwned[winningAddresses[i]] = _rOwned[winningAddresses[i]].add(eachPortion);
960
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 958

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
957   uint256 eachPortion = _rOwned[_moonWalletAddress].div(_numberOfFirstPrizeWinner);
958   for (uint256 i = 0; i < winningAddresses.length; i++) {
959   _rOwned[winningAddresses[i]] = _rOwned[winningAddresses[i]].add(eachPortion);
960   _rOwned[winningAddresses[i]] = _rOwned[winningAddresses[i]].add(_jpPortion);
961   _rOwned[_moonJPAddress] = _rOwned[_moonJPAddress].sub(_jpPortion);
962
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 969

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
968   _maxTxAmount = _tTotal.mul(maxTxPercent).div(
969   10**2
970   );
971   }
972
973
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 1020

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1019   uint256 tSupply = _tTotal;
1020   for (uint256 i = 0; i < _excluded.length; i++) {
1021   if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return
(_rTotal, _tTotal);
1022   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
1023   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
1024
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1056

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1055   return _amount.mul(_taxFee).div(
1056   10**2
1057   );
1058   }
1059
1060
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1062

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1061   return _amount.mul(_developmentFee).div(
1062   10**2
1063   );
1064   }
1065
1066
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1068

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1067   return _amount.mul(_moonFee).div(
1068   10**2
1069   );
1070   }
1071
1072
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1074

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1073   return _amount.mul(_liquidityFee).div(
1074   10**2
1075   );
1076   }
1077
1078
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 1292

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1291   delete allNumbers;
1292   drawNo ++ ;
1293   uint256 moonSpecialNumber;
1294   uint256 randNonce = _rTotal.div(10**12);
1295
1296
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1294

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1293   uint256 moonSpecialNumber;
1294   uint256 randNonce = _rTotal.div(10**12);
1295
1296   for ( uint i = 1 ; i < 6 ; i++) {
1297   uint j;
1298
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 1296

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1295
1296   for ( uint i = 1 ; i < 6 ; i++) {
1297   uint j;
1298   uint countUnmatched;
1299
1300
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 1300

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1299
1300   randNonce++;
1301   j =  uint(keccak256(abi.encodePacked(block.timestamp,msg.sender, randNonce*(i-
1)))) % _modulus+1;
1302   if (i>1) {
1303   for (uint k = 0; k< allNumbers.length; k++) {
1304
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1301

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1300   randNonce++;
1301   j =  uint(keccak256(abi.encodePacked(block.timestamp,msg.sender, randNonce*(i-
1)))) % _modulus+1;
1302   if (i>1) {
1303   for (uint k = 0; k< allNumbers.length; k++) {
1304   if (allNumbers[k] != j) {
1305
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 1303

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1302   if (i>1) {
1303   for (uint k = 0; k< allNumbers.length; k++) {
1304   if (allNumbers[k] != j) {
1305   countUnmatched++ ;
1306   }
1307
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 1305

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1304   if (allNumbers[k] != j) {
1305   countUnmatched++ ;
1306   }
1307   while ( countUnmatched < allNumbers.length ) {
1308   countUnmatched = 0;
1309
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1309

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1308   countUnmatched = 0;
1309   j =uint(keccak256(abi.encodePacked(randNonce++))) % _modulus + 1;
1310   for (uint r = 0; r< allNumbers.length; r++) {
1311   if (allNumbers[r] != j) {
1312   countUnmatched++ ;
1313
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 1310

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1309   j =uint(keccak256(abi.encodePacked(randNonce++))) % _modulus + 1;
1310   for (uint r = 0; r< allNumbers.length; r++) {
1311   if (allNumbers[r] != j) {
1312   countUnmatched++ ;
1313   }
1314
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 1312

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1311   if (allNumbers[r] != j) {
1312   countUnmatched++ ;
1313   }
1314   }
1315   }
1316
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1321

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1320
1321   uint m = allNumbers.length - 1;
1322   while (m > 0) {
1323   if (allNumbers[m] > allNumbers[m-1]) {
1324   break;
1325
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1323

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1322   while (m > 0) {
1323   if (allNumbers[m] > allNumbers[m-1]) {
1324   break;
1325   }
1326   uint n = allNumbers[m-1];
1327
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1326

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1325   }
1326   uint n = allNumbers[m-1];
1327   allNumbers[m-1] = allNumbers[m];
1328   allNumbers[m] = n;
1329   m--;
1330
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1327

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1326   uint n = allNumbers[m-1];
1327   allNumbers[m-1] = allNumbers[m];
1328   allNumbers[m] = n;
1329   m--;
1330   }
1331
```

SWC-101 | ARITHMETIC OPERATION "--" DISCOVERED

LINE 1329

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1328   allNumbers[m] = n;
1329   m--;
1330   }
1331
1332   }
1333
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 1334

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1333
1334   randNonce++;
1335   moonSpecialNumber =  uint(keccak256(abi.encodePacked(block.timestamp,msg.sender,
randNonce*777))) % 8 + 1;
1336   allNumbers.push(moonSpecialNumber);
1337
1338
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1335

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1334   randNonce++;
1335   moonSpecialNumber =  uint(keccak256(abi.encodePacked(block.timestamp,msg.sender,
randNonce*777))) % 8 + 1;
1336   allNumbers.push(moonSpecialNumber);
1337
1338   allPastDraws[drawNo] = allNumbers;
1339
```

SWC-101 | COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED

LINE 901

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
900   if (_excluded[i] == account) {
901   _excluded[i] = _excluded[_excluded.length - 1];
902   _tOwned[account] = 0;
903   _isExcluded[account] = false;
904   _excluded.pop();
905
```

SWC-101 | COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED

LINE 1301

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1300   randNonce++;
1301   j =  uint(keccak256(abi.encodePacked(block.timestamp,msg.sender, randNonce*(i-
1)))) % _modulus+1;
1302   if (i>1) {
1303   for (uint k = 0; k< allNumbers.length; k++) {
1304   if (allNumbers[k] != j) {
1305
```

SWC-101 | COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED

LINE 1321

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1320
1321   uint m = allNumbers.length - 1;
1322   while (m > 0) {
1323   if (allNumbers[m] > allNumbers[m-1]) {
1324   break;
1325
```

SWC-101 | COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED

LINE 1323

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1322   while (m > 0) {
1323   if (allNumbers[m] > allNumbers[m-1]) {
1324   break;
1325   }
1326   uint n = allNumbers[m-1];
1327
```

SWC-101 | COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED

LINE 1326

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1325   }
1326   uint n = allNumbers[m-1];
1327   allNumbers[m-1] = allNumbers[m];
1328   allNumbers[m] = n;
1329   m--;
1330
```

SWC-101 | COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED

LINE 1327

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1326   uint n = allNumbers[m-1];
1327   allNumbers[m-1] = allNumbers[m];
1328   allNumbers[m] = n;
1329   m--;
1330   }
1331
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 5

low SEVERITY

The current pragma Solidity directive is ""^0.8.4"". 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

- SuperMoonLotto.sol

Locations

```
4
5   pragma solidity ^0.8.4;
6   // SPDX-License-Identifier: Unlicensed
7   interface IERC20 {
8
9
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 729

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
728   uint256 public _numberOfSecondPrizeWinner;
729   uint256[] allNumbers ;
730
731   IUniswapV2Router02 public uniswapV2Router;
732   address public uniswapV2Pair;
733
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 734

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
733
734   bool inSwapAndLiquify;
735   bool public swapAndLiquifyEnabled = true;   //CHANGE_OPTIONAL: enable / disable
locking `liquidityFee` to Pancakeswap
736   bool ownerInTransact = false;
737   bool devInTransact = false;
738
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 736

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
735   bool public swapAndLiquifyEnabled = true;   //CHANGE_OPTIONAL: enable / disable
locking `liquidityFee` to Pancakeswap
736   bool ownerInTransact = false;
737   bool devInTransact = false;
738   bool moonJpInTransact = false;
739   bool moonFeeInTransact = false;
740
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 737

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
736   bool ownerInTransact = false;
737   bool devInTransact = false;
738   bool moonJpInTransact = false;
739   bool moonFeeInTransact = false;
740
741
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 738

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
737   bool devInTransact = false;
738   bool moonJpInTransact = false;
739   bool moonFeeInTransact = false;
740
741   uint256 public _maxTxAmount = 20000 * 10**6 * 10**9;
//CHANGE_OPTIONAL: max amount of tokens that can be transferred per transaction
742
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 739

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
738   bool moonJpInTransact = false;
739   bool moonFeeInTransact = false;
740
741   uint256 public _maxTxAmount = 20000 * 10**6 * 10**9;
//CHANGE_OPTIONAL: max amount of tokens that can be transferred per transaction
742   uint256 private numTokensSellToAddToLiquidity = 20000 * 10**6 * 10**9;
//CHANGE_OPTIONAL: minimum number of tokens in contract to be sent to Pancakeswap pool
743
```

SWC-110 | PUBLIC STATE VARIABLE WITH ARRAY TYPE CAUSING REACHABLE EXCEPTION BY DEFAULT.

LINE 727

low SEVERITY

The public state variable "allPastDraws" in "SuperMoonLotto" contract has type "mapping(uint256 => uint256[])" and can cause an exception in case of use of invalid array index value.

Source File

- SuperMoonLotto.sol

Locations

```
726   uint256 public drawNo;
727   mapping (uint256 => uint256 []) public allPastDraws;
728   uint256 public _numberOfSecondPrizeWinner;
729   uint256[] allNumbers ;
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 900

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
899   for (uint256 i = 0; i < _excluded.length; i++) {
900   if (_excluded[i] == account) {
901   _excluded[i] = _excluded[_excluded.length - 1];
902   _tOwned[account] = 0;
903   _isExcluded[account] = false;
904
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 901

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
900   if (_excluded[i] == account) {
901   _excluded[i] = _excluded[_excluded.length - 1];
902   _tOwned[account] = 0;
903   _isExcluded[account] = false;
904   _excluded.pop();
905
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 950

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
949   for (uint256 i = 0; i < winningAddresses.length; i++) {
950   _rOwned[winningAddresses[i]] = _rOwned[winningAddresses[i]].add(eachPortion);
951   }
952   _rOwned[_moonWalletAddress] =
_rOwned[_moonWalletAddress].sub(_rOwned[_moonWalletAddress].div(10));
953   }
954
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 959

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
958   for (uint256 i = 0; i < winningAddresses.length; i++) {
959   _rOwned[winningAddresses[i]] = _rOwned[winningAddresses[i]].add(eachPortion);
960   _rOwned[winningAddresses[i]] = _rOwned[winningAddresses[i]].add(_jpPortion);
961   _rOwned[_moonJPAddress] = _rOwned[_moonJPAddress].sub(_jpPortion);
962   }
963
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 960

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
959   _rOwned[winningAddresses[i]] = _rOwned[winningAddresses[i]].add(eachPortion);
960   _rOwned[winningAddresses[i]] = _rOwned[winningAddresses[i]].add(_jpPortion);
961   _rOwned[_moonJPAddress] = _rOwned[_moonJPAddress].sub(_jpPortion);
962   }
963   _rOwned[_moonWalletAddress] =
_rOwned[_moonWalletAddress].sub(_rOwned[_moonWalletAddress]);
964
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1021

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1020   for (uint256 i = 0; i < _excluded.length; i++) {
1021   if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return
(_rTotal, _tTotal);
1022   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
1023   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
1024   }
1025
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1022

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1021   if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return
(_rTotal, _tTotal);
1022   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
1023   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
1024   }
1025   if (rSupply < _rTotal.div(_tTotal)) return (_rTotal, _tTotal);
1026
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1023

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1022   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
1023   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
1024   }
1025   if (rSupply < _rTotal.div(_tTotal)) return (_rTotal, _tTotal);
1026   return (rSupply, tSupply);
1027
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1204

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1203   address[] memory path = new address[](2);
1204   path[0] = address(this);
1205   path[1] = uniswapV2Router.WETH();
1206
1207   _approve(address(this), address(uniswapV2Router), tokenAmount);
1208
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1205

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1204   path[0] = address(this);
1205   path[1] = uniswapV2Router.WETH();
1206
1207   _approve(address(this), address(uniswapV2Router), tokenAmount);
1208
1209
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1304

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1303   for (uint k = 0; k< allNumbers.length; k++) {
1304   if (allNumbers[k] != j) {
1305   countUnmatched++ ;
1306   }
1307   while ( countUnmatched < allNumbers.length ) {
1308
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1311

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1310   for (uint r = 0; r< allNumbers.length; r++) {
1311   if (allNumbers[r] != j) {
1312   countUnmatched++ ;
1313   }
1314   }
1315
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1323

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1322   while (m > 0) {
1323   if (allNumbers[m] > allNumbers[m-1]) {
1324   break;
1325   }
1326   uint n = allNumbers[m-1];
1327
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1326

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1325   }
1326   uint n = allNumbers[m-1];
1327   allNumbers[m-1] = allNumbers[m];
1328   allNumbers[m] = n;
1329   m--;
1330
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1327

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1326   uint n = allNumbers[m-1];
1327   allNumbers[m-1] = allNumbers[m];
1328   allNumbers[m] = n;
1329   m--;
1330   }
1331
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1328

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1327   allNumbers[m-1] = allNumbers[m];
1328   allNumbers[m] = n;
1329   m--;
1330   }
1331
1332
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1341

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1340   emit NewDraw(
1341   allNumbers[0],
1342   allNumbers[1],
1343   allNumbers[2],
1344   allNumbers[3],
1345
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1342

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1341   allNumbers[0],
1342   allNumbers[1],
1343   allNumbers[2],
1344   allNumbers[3],
1345   allNumbers[4],
1346
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1343

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1342   allNumbers[1],
1343   allNumbers[2],
1344   allNumbers[3],
1345   allNumbers[4],
1346   moonSpecialNumber,
1347
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1344

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1343   allNumbers[2],
1344   allNumbers[3],
1345   allNumbers[4],
1346   moonSpecialNumber,
1347   drawNo
1348
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1345

low SEVERITY

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

Source File

- SuperMoonLotto.sol

Locations

```
1344   allNumbers[3],
1345   allNumbers[4],
1346   moonSpecialNumber,
1347   drawNo
1348   );
1349
```

DISCLAIMER

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to, or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without Sysfixed's prior written consent in each instance.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Sysfixed to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model, or legal compliance.

This is a limited report on our findings based on our analysis, in accordance with good industry practice as of the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

This report is provided for information purposes only and on a non-reliance basis and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and Sysfixed and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers, and other representatives) (Sysfixed) owe no duty of care.

ABOUT US

Sysfixed is a blockchain security certification organization established in 2021 with the objective to provide smart contract security services and verify their correctness in blockchain-based protocols. Sysfixed automatically scans for security vulnerabilities in Ethereum and other EVM-based blockchain smart contracts. Sysfixed a comprehensive range of analysis techniques—including static analysis, dynamic analysis, and symbolic execution—can accurately detect security vulnerabilities to provide an in-depth analysis report. With a vibrant ecosystem of world-class integration partners that amplify developer productivity, Sysfixed can be utilized in all phases of your project's lifecycle. Our team of security experts is dedicated to the research and improvement of our tools and techniques used to fortify your code.