MessaageBit

Smart Contract Audit Report

SYSFIXED

11 Jan 2023

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
MessaageBit	MB	Binance Smart Chain

Addresses

Contract address	0xb6e751BDD09d2d91452A1082D5D0f31cCd260924
Contract deployer address	0xBDfF438acc08065daB973B088e4dd7af7f0C45A4

Project Website

https://www.messaagebit.com/

Codebase

https://bscscan.com/address/0xb6e751BDD09d2d91452A1082D5D0f31cCd260924#code

SUMMARY

MessaageBit provides Messages App, crypto analytics and portfolio management for the BNB Chain tokens and aims to improve crypto Apps experience. Comming products: - Bridge - App - Swap - Lending Dapps: - CMC & CG - Big call & AMA groups been onboarded. Onder, Caesar, Gollum, VatorCapital, Cryptocat, Bossy, Future Lounge, Procify Ads, DefiApetalk, Sherlock, Shadowcall, HulkGems, Phoniex, Doxxed Always, PythagorasDev

Contract Summary

Documentation Quality

MessaageBit 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 MessaageBit 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 737.
- SWC-101 | It is recommended to use vetted safe math libraries for arithmetic operations consistently on lines 125, 157, 180, 181, 216, 252, 479, 720, 720, 720, 720, 721, 721, 740, 740, 740, 740, 741, 741, 741, 741, 872, 874, 911, 957, 976, 982 and 874.
- SWC-103 | Pragma statements can be allowed to float when a contract is intended on lines 26.
- SWC-110 SWC-123 | It is recommended to use of revert(), assert(), and require() in Solidity, and the new REVERT opcode in the EVM on lines 873, 874, 874, 958, 958, 959, 960, 1085 and 1086.

CONCLUSION

We have audited the MessaageBit project released on January 2023 to discover issues and identify potential security vulnerabilities in MessaageBit 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 MessaageBit smart contract code 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 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 SWC-123	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	Tuesday Jan 10 2023 03:23:11 GMT+0000 (Coordinated Universal Time)
Finished	Wednesday Jan 11 2023 08:18:58 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	MessaageBit.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

SYSFIXED

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-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-108	STATE VARIABLE VISIBILITY IS NOT SET.	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 125

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
124   function add(uint256 a, uint256 b) internal pure returns (uint256) {
125   uint256 c = a + b;
126   require(c >= a, "SafeMath: addition overflow");
127
128   return c;
129
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 157

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
156   require(b <= a, errorMessage);
157   uint256 c = a - b;
158
159   return c;
160   }
161
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 180

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
179
180   uint256 c = a * b;
181   require(c / a == b, "SafeMath: multiplication overflow");
182
183   return c;
184
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 181

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
180   uint256 c = a * b;
181   require(c / a == b, "SafeMath: multiplication overflow");
182
183   return c;
184   }
185
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 216

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
215   require(b > 0, errorMessage);
216   uint256 c = a / b;
217   // assert(a == b * c + a % b); // There is no case in which this doesn't hold
218
219   return c;
220
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 252

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
251   require(b != 0, errorMessage);
252   return a % b;
253   }
254   }
255
256
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 479

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
478   _owner = address(0);
479   _lockTime = now + time;
480   emit OwnershipTransferred(_owner, address(0));
481   }
482
483
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 720

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
719   uint256 private constant MAX = ~uint256(0);
720   uint256 private _tTotal = 1000000000 * 10**6 * 10**9;
721   uint256 private _rTotal = (MAX - (MAX % _tTotal));
722   uint256 private _tFeeTotal;
723
724
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 720

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
uint256 private constant MAX = ~uint256(0);
uint256 private _tTotal = 1000000000 * 10**6 * 10**9;
uint256 private _rTotal = (MAX - (MAX % _tTotal));
uint256 private _tFeeTotal;


```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 720

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
719   uint256 private constant MAX = ~uint256(0);
720   uint256 private _tTotal = 1000000000 * 10**6 * 10**9;
721   uint256 private _rTotal = (MAX - (MAX % _tTotal));
722   uint256 private _tFeeTotal;
723
724
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 720

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
719   uint256 private constant MAX = ~uint256(0);
720   uint256 private _tTotal = 1000000000 * 10**6 * 10**9;
721   uint256 private _rTotal = (MAX - (MAX % _tTotal));
722   uint256 private _tFeeTotal;
723
724
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 721

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
720   uint256 private _tTotal = 1000000000 * 10**6 * 10**9;
721   uint256 private _rTotal = (MAX - (MAX % _tTotal));
722   uint256 private _tFeeTotal;
723
724   string private _name = "MessaageBit";
725
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 721

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
720   uint256 private _tTotal = 1000000000 * 10**6 * 10**9;
721   uint256 private _rTotal = (MAX - (MAX % _tTotal));
722   uint256 private _tFeeTotal;
723
724   string private _name = "MessaageBit";
725
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 740

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
739
740    uint256 public _maxTxAmount = 5000000 * 10**6 * 10**9;
741    uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**9;
742
743    event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);
744
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 740

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
739
740   uint256 public _maxTxAmount = 5000000 * 10**6 * 10**9;
741   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**9;
742
743   event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);
744
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 740

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
739
740   uint256 public _maxTxAmount = 5000000 * 10**6 * 10**9;
741   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**9;
742
743   event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);
744
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 740

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
739
740   uint256 public _maxTxAmount = 5000000 * 10**6 * 10**9;
741   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**9;
742
743   event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);
744
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 741

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
740   uint256 public _maxTxAmount = 5000000 * 10**6 * 10**9;
741   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**9;
742
743   event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);
744   event SwapAndLiquifyEnabledUpdated(bool enabled);
745
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 741

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
740   uint256 public _maxTxAmount = 5000000 * 10**6 * 10**9;
741   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**9;
742
743   event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);
744   event SwapAndLiquifyEnabledUpdated(bool enabled);
745
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 741

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
740   uint256 public _maxTxAmount = 5000000 * 10**6 * 10**9;
741   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**9;
742
743   event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);
744   event SwapAndLiquifyEnabledUpdated(bool enabled);
745
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 741

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
740   uint256 public _maxTxAmount = 5000000 * 10**6 * 10**9;
741   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**9;
742
743   event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);
744   event SwapAndLiquifyEnabledUpdated(bool enabled);
745
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 872

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
871   require(_isExcluded[account], "Account is already excluded");
872   for (uint256 i = 0; i < _excluded.length; i++) {
873   if (_excluded[i] == account) {
874   _excluded[i] = _excluded[_excluded.length - 1];
875   _tOwned[account] = 0;
876
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 874

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
873   if (_excluded[i] == account) {
874   _excluded[i] = _excluded[_excluded.length - 1];
875   _tOwned[account] = 0;
876   _isExcluded[account] = false;
877   _excluded.pop();
878
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 911

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
910   _maxTxAmount = _tTotal.mul(maxTxPercent).div(
911   10**2
912   );
913   }
914
915
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 957

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
956   uint256 tSupply = _tTotal;
957   for (uint256 i = 0; i < _excluded.length; i++) {
958   if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return
(_rTotal, _tTotal);
959   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
960   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
961
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 976

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
975   return _amount.mul(_taxFee).div(
976   10**2
977   );
978   }
979
980
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 982

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
981   return _amount.mul(_liquidityFee).div(
982   10**2
983   );
984   }
985
986
```

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

LINE 874

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
873   if (_excluded[i] == account) {
874   _excluded[i] = _excluded[_excluded.length - 1];
875   _tOwned[account] = 0;
876   _isExcluded[account] = false;
877   _excluded.pop();
878
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 26

low SEVERITY

The current pragma Solidity directive is ""^0.6.12"". 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

- MessaageBit.sol

Locations

```
25
26   pragma solidity ^0.6.12;
27   // SPDX-License-Identifier: Unlicensed
28   interface IERC20 {
29
30
```

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 "inSwapAndLiquify" is internal. Other possible visibility settings are public and private.

Source File

- MessaageBit.sol

Locations

```
736
737    bool inSwapAndLiquify;
738    bool public swapAndLiquifyEnabled = true;
739
740    uint256 public _maxTxAmount = 5000000 * 10**6 * 10**9;
741
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 873

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
872   for (uint256 i = 0; i < _excluded.length; i++) {
873   if (_excluded[i] == account) {
874   _excluded[i] = _excluded[_excluded.length - 1];
875   _tOwned[account] = 0;
876   _isExcluded[account] = false;
877
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 874

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
873   if (_excluded[i] == account) {
874   _excluded[i] = _excluded[_excluded.length - 1];
875   _tOwned[account] = 0;
876   _isExcluded[account] = false;
877   _excluded.pop();
878
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 874

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
873   if (_excluded[i] == account) {
874   _excluded[i] = _excluded[_excluded.length - 1];
875   _tOwned[account] = 0;
876   _isExcluded[account] = false;
877   _excluded.pop();
878
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 958

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
957   for (uint256 i = 0; i < _excluded.length; i++) {
958   if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return
(_rTotal, _tTotal);
959   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
960   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
961   }
962
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 958

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
957   for (uint256 i = 0; i < _excluded.length; i++) {
958   if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return
(_rTotal, _tTotal);
959   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
960   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
961   }
962
```

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

- MessaageBit.sol

Locations

```
958   if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return
(_rTotal, _tTotal);
959   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
960   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
961   }
962   if (rSupply < _rTotal.div(_tTotal)) return (_rTotal, _tTotal);
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

- MessaageBit.sol

Locations

```
959   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
960   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
961   }
962   if (rSupply < _rTotal.div(_tTotal)) return (_rTotal, _tTotal);
963   return (rSupply, tSupply);
964
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1085

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
1084   address[] memory path = new address[](2);
1085   path[0] = address(this);
1086   path[1] = uniswapV2Router.WETH();
1087
1088   _approve(address(this), address(uniswapV2Router), tokenAmount);
1089
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1086

low SEVERITY

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

Source File

- MessaageBit.sol

Locations

```
1085    path[0] = address(this);
1086    path[1] = uniswapV2Router.WETH();
1087
1088    _approve(address(this), address(uniswapV2Router), tokenAmount);
1089
1090
```

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.