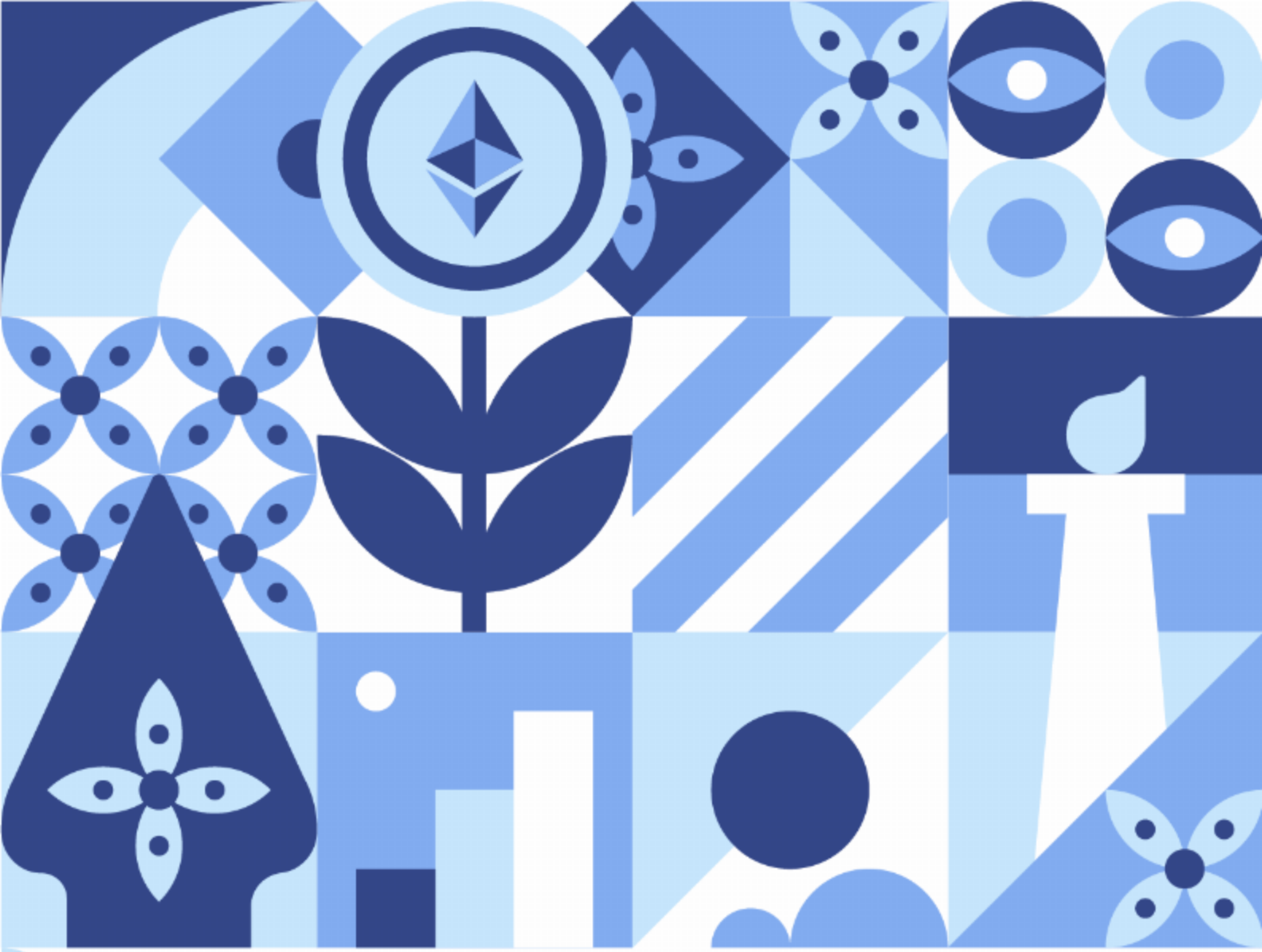

ROBO INU

Smart Contract Audit Report

SYSFIXED

08 Nov 2021

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
ROBO INU	RBIF	Ethereum

Addresses

Contract address	0x7b32e70e8d73ac87c1b342e063528b2930b15ceb
Contract deployer address	0xa116abd1B09c0b32B60260E2a65A3063AFf78B75

Project Website

https://t.me/metagochi

Codebase

https://etherscan.io/address/0x7b32e70e8d73ac87c1b342e063528b2930b15ceb#code

SUMMARY

ROBO INU FINANCE is one of the many financial projects being spearheaded by ROBO GLOBAL INVESTMENT PTE LTD. Beyond the individuals, ROBO GLOBAL INVESTMENT aspires to create an open ecosystem where anyone, including you, can gain financial freedom. ROBO INU FINANCE leverages on blockchain technology to enhance the lives of individuals and business operations.

Contract Summary

Documentation Quality

ROBO INU 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 ROBO INU 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 577.
- SWC-101 | It is recommended to use vetted safe math libraries for arithmetic operations consistently on lines 55, 65, 76, 77, 86, 94, 100, 105, 110, 115, 120, 131, 143, 155, 546, 546, 546, 546, 547, 547, 573, 573, 573, 573, 574, 574, 574, 574, 575, 575, 575, 575, 721, 723, 945, 964, 970, 976, 1112, 1112, 1112, 1112, 1134, 1134, 1134, 1134 and 723.
- SWC-103 | Pragma statements can be allowed to float when a contract is intended on lines 18.
- 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 722, 723, 723, 799, 800, 946, 946, 947 and 948.

CONCLUSION

We have audited the ROBO INU project released on November 2022 to discover issues and identify potential security vulnerabilities in ROBO INU 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 ROBO INU 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
Unprotected Ether Withdrawal	SWC-105	Due to missing or insufficient access controls, malicious parties can withdraw from the contract.	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
Uninitialized Storage Pointer	SWC-109	Uninitialized local storage variables can point to unexpected storage locations in the contract.	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	Delegatecalls 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
Incorrect Constructor Name	SWC-118	Constructors are special functions that are called only once during the contract creation.	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
Write to Arbitrary Storage Location	SWC-124	The contract is responsible for ensuring that only authorized user or contract accounts may write to sensitive storage locations.	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
Insufficient Gas Griefing	SWC-126	Insufficient gas griefing attacks can be performed on contracts which accept data and use it in a sub-call on another contract.	PASS
Arbitrary Jump Function	SWC-127	As Solidity doesnt support pointer arithmetics, it is impossible to change such variable to an arbitrary value.	PASS

Typographical Error	SWC-129	A typographical error can occur for example when the intent of a defined operation is to sum a number to a variable.	PASS
Override control character	SWC-130	Malicious actors can use the Right-To-Left-Override unicode character to force RTL text rendering and confuse users as to the real intent of a contract.	PASS
Unused variables	SWC-131 SWC-135	Unused variables are allowed in Solidity and they do not pose a direct security issue.	PASS
Unexpected Ether balance	SWC-132	Contracts can behave erroneously when they strictly assume a specific Ether balance.	PASS
Hash Collisions Variable	SWC-133	Using abi.encodePacked() with multiple variable length arguments can, in certain situations, lead to a hash collision.	PASS
Hardcoded gas amount	SWC-134	The transfer() and send() functions forward a fixed amount of 2300 gas.	PASS
Unencrypted Private Data	SWC-136	It is a common misconception that private type variables cannot be read.	PASS

SMART CONTRACT ANALYSIS

Started	Sunday Nov 07 2021 10:40:50 GMT+0000 (Coordinated Universal Time)
Finished	Monday Nov 08 2021 14:29:12 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	Token.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	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-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 55

low SEVERITY

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

Source File

- Token.sol

Locations

```
54   unchecked {
55   uint256 c = a + b;
56   if (c < a) return (false, 0);
57   return (true, c);
58   }
59
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 65

low SEVERITY

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

Source File

- Token.sol

Locations

```
64   if (b > a) return (false, 0);
65   return (true, a - b);
66   }
67   }
68
69
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 76

low SEVERITY

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

Source File

- Token.sol

Locations

```
75   if (a == 0) return (true, 0);
76   uint256 c = a * b;
77   if (c / a != b) return (false, 0);
78   return (true, c);
79   }
80
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 77

low SEVERITY

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

Source File

- Token.sol

Locations

```
76   uint256 c = a * b;
77   if (c / a != b) return (false, 0);
78   return (true, c);
79   }
80   }
81
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 86

low SEVERITY

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

Source File

- Token.sol

Locations

```
85   if (b == 0) return (false, 0);
86   return (true, a / b);
87   }
88   }
89
90
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 94

low SEVERITY

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

Source File

- Token.sol

Locations

```
93   if (b == 0) return (false, 0);
94   return (true, a % b);
95   }
96   }
97
98
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 100

low SEVERITY

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

Source File

- Token.sol

Locations

```
99   function add(uint256 a, uint256 b) internal pure returns (uint256) {
100   return a + b;
101   }
102
103
104
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 105

low SEVERITY

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

Source File

- Token.sol

Locations

```
104   function sub(uint256 a, uint256 b) internal pure returns (uint256) {
105   return a - b;
106   }
107
108
109
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 110

low SEVERITY

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

Source File

- Token.sol

Locations

```
109   function mul(uint256 a, uint256 b) internal pure returns (uint256) {
110   return a * b;
111   }
112
113
114
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 115

low SEVERITY

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

Source File

- Token.sol

Locations

```
114   function div(uint256 a, uint256 b) internal pure returns (uint256) {
115   return a / b;
116   }
117
118
119
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 120

low SEVERITY

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

Source File

- Token.sol

Locations

```
119   function mod(uint256 a, uint256 b) internal pure returns (uint256) {
120   return a % b;
121   }
122
123
124
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 131

low SEVERITY

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

Source File

- Token.sol

Locations

```
130   require(b <= a, errorMessage);
131   return a - b;
132   }
133   }
134
135
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 143

low SEVERITY

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

Source File

- Token.sol

Locations

```
142   require(b > 0, errorMessage);
143   return a / b;
144   }
145   }
146
147
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 155

low SEVERITY

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

Source File

- Token.sol

Locations

```
154   require(b > 0, errorMessage);
155   return a % b;
156   }
157   }
158   }
159
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 546

low SEVERITY

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

Source File

- Token.sol

Locations

```
545   uint256 private constant MAX = ~uint256(0);
546   uint256 private _tTotal = 100000000000 * 10**6 * 10**9;
547   uint256 private _rTotal = (MAX - (MAX % _tTotal));
548   uint256 private _tFeeTotal;
549
550
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 546

low SEVERITY

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

Source File

- Token.sol

Locations

```
545   uint256 private constant MAX = ~uint256(0);
546   uint256 private _tTotal = 100000000000 * 10**6 * 10**9;
547   uint256 private _rTotal = (MAX - (MAX % _tTotal));
548   uint256 private _tFeeTotal;
549
550
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 546

low SEVERITY

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

Source File

- Token.sol

Locations

```
545   uint256 private constant MAX = ~uint256(0);
546   uint256 private _tTotal = 100000000000 * 10**6 * 10**9;
547   uint256 private _rTotal = (MAX - (MAX % _tTotal));
548   uint256 private _tFeeTotal;
549
550
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 546

low SEVERITY

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

Source File

- Token.sol

Locations

```
545   uint256 private constant MAX = ~uint256(0);
546   uint256 private _tTotal = 100000000000 * 10**6 * 10**9;
547   uint256 private _rTotal = (MAX - (MAX % _tTotal));
548   uint256 private _tFeeTotal;
549
550
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 547

low SEVERITY

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

Source File

- Token.sol

Locations

```
546   uint256 private _tTotal = 100000000000 * 10**6 * 10**9;
547   uint256 private _rTotal = (MAX - (MAX % _tTotal));
548   uint256 private _tFeeTotal;
549
550   string private _name = "ROBO INU";
551
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 547

low SEVERITY

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

Source File

- Token.sol

Locations

```
546   uint256 private _tTotal = 100000000000 * 10**6 * 10**9;
547   uint256 private _rTotal = (MAX - (MAX % _tTotal));
548   uint256 private _tFeeTotal;
549
550   string private _name = "ROBO INU";
551
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 573

low SEVERITY

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

Source File

- Token.sol

Locations

```
572
573   uint256 public _maxTxAmount = 300000000 * 10**6 * 10**9;
574   uint256 private numTokensSellToAddToLiquidity = 20000000 * 10**6 * 10**9;
575   uint256 public _maxTokenHolder = 2000000000 * 10**6 * 10**9;
576
577
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 573

low SEVERITY

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

Source File

- Token.sol

Locations

```
572
573   uint256 public _maxTxAmount = 300000000 * 10**6 * 10**9;
574   uint256 private numTokensSellToAddToLiquidity = 20000000 * 10**6 * 10**9;
575   uint256 public _maxTokenHolder = 2000000000 * 10**6 * 10**9;
576
577
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 573

low SEVERITY

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

Source File

- Token.sol

Locations

```
572
573   uint256 public _maxTxAmount = 300000000 * 10**6 * 10**9;
574   uint256 private numTokensSellToAddToLiquidity = 20000000 * 10**6 * 10**9;
575   uint256 public _maxTokenHolder = 2000000000 * 10**6 * 10**9;
576
577
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 573

low SEVERITY

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

Source File

- Token.sol

Locations

```
572
573   uint256 public _maxTxAmount = 300000000 * 10**6 * 10**9;
574   uint256 private numTokensSellToAddToLiquidity = 20000000 * 10**6 * 10**9;
575   uint256 public _maxTokenHolder = 2000000000 * 10**6 * 10**9;
576
577
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 574

low SEVERITY

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

Source File

- Token.sol

Locations

```
573   uint256 public _maxTxAmount = 300000000 * 10**6 * 10**9;
574   uint256 private numTokensSellToAddToLiquidity = 20000000 * 10**6 * 10**9;
575   uint256 public _maxTokenHolder = 2000000000 * 10**6 * 10**9;
576
577   bool inSwapAndLiquify;
578
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 574

low SEVERITY

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

Source File

- Token.sol

Locations

```
573   uint256 public _maxTxAmount = 300000000 * 10**6 * 10**9;
574   uint256 private numTokensSellToAddToLiquidity = 20000000 * 10**6 * 10**9;
575   uint256 public _maxTokenHolder = 2000000000 * 10**6 * 10**9;
576
577   bool inSwapAndLiquify;
578
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 574

low SEVERITY

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

Source File

- Token.sol

Locations

```
573   uint256 public _maxTxAmount = 300000000 * 10**6 * 10**9;
574   uint256 private numTokensSellToAddToLiquidity = 20000000 * 10**6 * 10**9;
575   uint256 public _maxTokenHolder = 2000000000 * 10**6 * 10**9;
576
577   bool inSwapAndLiquify;
578
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 574

low SEVERITY

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

Source File

- Token.sol

Locations

```
573   uint256 public _maxTxAmount = 300000000 * 10**6 * 10**9;
574   uint256 private numTokensSellToAddToLiquidity = 20000000 * 10**6 * 10**9;
575   uint256 public _maxTokenHolder = 2000000000 * 10**6 * 10**9;
576
577   bool inSwapAndLiquify;
578
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 575

low SEVERITY

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

Source File

- Token.sol

Locations

```
574   uint256 private numTokensSellToAddToLiquidity = 20000000 * 10**6 * 10**9;
575   uint256 public _maxTokenHolder = 2000000000 * 10**6 * 10**9;
576
577   bool inSwapAndLiquify;
578   bool public swapAndLiquifyEnabled = false;
579
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 575

low SEVERITY

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

Source File

- Token.sol

Locations

```
574   uint256 private numTokensSellToAddToLiquidity = 20000000 * 10**6 * 10**9;
575   uint256 public _maxTokenHolder = 2000000000 * 10**6 * 10**9;
576
577   bool inSwapAndLiquify;
578   bool public swapAndLiquifyEnabled = false;
579
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 575

low SEVERITY

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

Source File

- Token.sol

Locations

```
574   uint256 private numTokensSellToAddToLiquidity = 20000000 * 10**6 * 10**9;
575   uint256 public _maxTokenHolder = 2000000000 * 10**6 * 10**9;
576
577   bool inSwapAndLiquify;
578   bool public swapAndLiquifyEnabled = false;
579
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 575

low SEVERITY

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

Source File

- Token.sol

Locations

```
574   uint256 private numTokensSellToAddToLiquidity = 20000000 * 10**6 * 10**9;
575   uint256 public _maxTokenHolder = 2000000000 * 10**6 * 10**9;
576
577   bool inSwapAndLiquify;
578   bool public swapAndLiquifyEnabled = false;
579
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 721

low SEVERITY

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

Source File

- Token.sol

Locations

```
720   require(_excluded.length < 20, "Excluded list too big");
721   for (uint256 i = 0; i < _excluded.length; i++) {
722   if (_excluded[i] == account) {
723   _excluded[i] = _excluded[_excluded.length - 1];
724   _tOwned[account] = 0;
725
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 723

low SEVERITY

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

Source File

- Token.sol

Locations

```
722   if (_excluded[i] == account) {
723   _excluded[i] = _excluded[_excluded.length - 1];
724   _tOwned[account] = 0;
725   _isExcluded[account] = false;
726   _excluded.pop();
727
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 945

low SEVERITY

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

Source File

- Token.sol

Locations

```
944   uint256 tSupply = _tTotal;
945   for (uint256 i = 0; i < _excluded.length; i++) {
946   if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return
(_rTotal, _tTotal);
947   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
948   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
949
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 964

low SEVERITY

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

Source File

- Token.sol

Locations

```
963   return _amount.mul(_taxFee).div(
964   10**2
965   );
966   }
967
968
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 970

low SEVERITY

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

Source File

- Token.sol

Locations

```
969   return _amount.mul(_liquidityFee).div(
970   10**2
971   );
972   }
973
974
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 976

low SEVERITY

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

Source File

- Token.sol

Locations

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

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1112

low SEVERITY

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

Source File

- Token.sol

Locations

```
1111
1112   _maxTxAmount = 1000000000 * 10**6 * 10**9;
1113   }
1114
1115   function goLive() external onlyOwner {
1116
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1112

low SEVERITY

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

Source File

- Token.sol

Locations

```
1111
1112   _maxTxAmount = 1000000000 * 10**6 * 10**9;
1113   }
1114
1115   function goLive() external onlyOwner {
1116
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1112

low SEVERITY

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

Source File

- Token.sol

Locations

```
1111
1112   _maxTxAmount = 1000000000 * 10**6 * 10**9;
1113   }
1114
1115   function goLive() external onlyOwner {
1116
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1112

low SEVERITY

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

Source File

- Token.sol

Locations

```
1111
1112   _maxTxAmount = 1000000000 * 10**6 * 10**9;
1113   }
1114
1115   function goLive() external onlyOwner {
1116
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1134

low SEVERITY

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

Source File

- Token.sol

Locations

```
1133
1134   _maxTxAmount = 300000000 * 10**6 * 10**9;
1135   }
1136
1137   }
1138
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1134

low SEVERITY

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

Source File

- Token.sol

Locations

```
1133
1134   _maxTxAmount = 300000000 * 10**6 * 10**9;
1135   }
1136
1137   }
1138
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1134

low SEVERITY

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

Source File

- Token.sol

Locations

```
1133
1134   _maxTxAmount = 300000000 * 10**6 * 10**9;
1135   }
1136
1137   }
1138
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1134

low SEVERITY

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

Source File

- Token.sol

Locations

```
1133
1134   _maxTxAmount = 300000000 * 10**6 * 10**9;
1135   }
1136
1137   }
1138
```

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

LINE 723

low SEVERITY

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

Source File

- Token.sol

Locations

```
722   if (_excluded[i] == account) {
723   _excluded[i] = _excluded[_excluded.length - 1];
724   _tOwned[account] = 0;
725   _isExcluded[account] = false;
726   _excluded.pop();
727
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 18

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

- Token.sol

Locations

```
17   **/
18   pragma solidity ^0.8.4;
19
20
21   interface IERC20 {
22
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 577

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

- Token.sol

Locations

```
576
577   bool inSwapAndLiquify;
578   bool public swapAndLiquifyEnabled = false;
579
580   event SwapAndLiquifyEnabledUpdated(bool enabled);
581
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 722

low SEVERITY

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

Source File

- Token.sol

Locations

```
721   for (uint256 i = 0; i < _excluded.length; i++) {
722   if (_excluded[i] == account) {
723   _excluded[i] = _excluded[_excluded.length - 1];
724   _tOwned[account] = 0;
725   _isExcluded[account] = false;
726
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 723

low SEVERITY

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

Source File

- Token.sol

Locations

```
722   if (_excluded[i] == account) {
723   _excluded[i] = _excluded[_excluded.length - 1];
724   _tOwned[account] = 0;
725   _isExcluded[account] = false;
726   _excluded.pop();
727
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 723

low SEVERITY

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

Source File

- Token.sol

Locations

```
722   if (_excluded[i] == account) {
723   _excluded[i] = _excluded[_excluded.length - 1];
724   _tOwned[account] = 0;
725   _isExcluded[account] = false;
726   _excluded.pop();
727
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 799

low SEVERITY

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

Source File

- Token.sol

Locations

```
798   address[] memory path = new address[](2);
799   path[0] = address(this);
800   path[1] = uniswapV2Router.WETH();
801
802   _approve(address(this), address(uniswapV2Router), tokenAmount);
803
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 800

low SEVERITY

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

Source File

- Token.sol

Locations

```
799   path[0] = address(this);
800   path[1] = uniswapV2Router.WETH();
801
802   _approve(address(this), address(uniswapV2Router), tokenAmount);
803
804
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 946

low SEVERITY

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

Source File

- Token.sol

Locations

```
945   for (uint256 i = 0; i < _excluded.length; i++) {
946   if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return
(_rTotal, _tTotal);
947   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
948   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
949   }
950
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 946

low SEVERITY

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

Source File

- Token.sol

Locations

```
945   for (uint256 i = 0; i < _excluded.length; i++) {
946   if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return
(_rTotal, _tTotal);
947   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
948   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
949   }
950
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 947

low SEVERITY

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

Source File

- Token.sol

Locations

```
946   if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return
(_rTotal, _tTotal);
947   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
948   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
949   }
950   if (rSupply < _rTotal.div(_tTotal)) return (_rTotal, _tTotal);
951
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 948

low SEVERITY

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

Source File

- Token.sol

Locations

```
947   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
948   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
949   }
950   if (rSupply < _rTotal.div(_tTotal)) return (_rTotal, _tTotal);
951   return (rSupply, tSupply);
952
```

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.