Flokimooni

Smart Contract Audit Report

SYSFIXED

14 Oct 2021

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
Flokimooni	Flokim	Binance Smart Chain

Addresses

Contract address	0x0f5351b9eaefd6687dff143de6ea5d01cb9c1205
Contract deployer address	0x069fD156c0d22E5D5F68e92f3237624B8eB6Ae9C

Project Website

http://www.flokimooni.com/

Codebase

https://bscscan.com/address/0x0f5351b9eaefd6687dff143de6ea5d01cb9c1205#code

SUMMARY

Flokimooni is named after Elon Musk's Shiba Inu. Born by fans and members of the Shiba Inu community, Flokimooni spread like a movement. Creating a beautiful ecosystem with the most modern features in the crypto space: MooniWorld. By combining the power of memes with real utility, Flokimooni aims to be a top 100 crypto project and plans to kickstart the next crypto revolution. We call our community the Flokimoonies. With the power of our dedicated team and loyal holders, Flokimooni can compete with any of the top tokens on the market.

Contract Summary

Documentation Quality

Flokimooni 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 Flokimooni with the discovery of several low issues.

Test Coverage

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

Audit Findings Summary

- SWC-101 | It is recommended to use vetted safe math libraries for arithmetic operations consistently on lines 138, 150, 163, 164, 175, 185, 199, 216, 231, 232, 250, 267, 285, 305, 325, 1005, 1017, 1030, 1031, 1042, 1052, 1066, 1083, 1098, 1099, 1117, 1134, 1152, 1172, 1192, 2060, 2064, 2076, 2083, 2092, 2183, 2314, 2349, 2411, 2564, 2659, 2853, 3161, 3171, 3175 and 2183.
- SWC-103 | Pragma statements can be allowed to float when a contract is intended on lines 2230.
- 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 2154, 2184, 2189, 2552, 2552, 2555, 2556, 2558, 2569, 2579, 2583, 2587, 2594, 2595, 2660, 2912, 2913, 2929, 2930, 2931 and 3167.
- SWC-115 | tx.origin should not be used for authorization, use msg.sender instead on lines 2789 and 2873.

CONCLUSION

We have audited the Flokimooni project released on January 2023 to discover issues and identify potential security vulnerabilities in Flokimooni 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 satisfactory results with low-risk issues.

The issues found in the Flokimooni 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, tx.origin as a part of authorization control, 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. Use of "tx.origin" as a part of authorization control, using "tx.origin" as a security control can lead to authorization bypass vulnerabilities. Consider using "msg.sender" unless you really know what you are doing.

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.	PASS
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.	ISSUE FOUND
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	Wednesday Oct 13 2021 14:38:29 GMT+0000 (Coordinated Universal Time)
Finished	Thursday Oct 14 2021 15:24:12 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	AntiBotBABYTOKEN.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	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-115	USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.	low	acknowledged
SWC-115	USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.	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

SYSFIXED

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 138

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
137   function tryAdd(uint256 a, uint256 b) internal pure returns (bool, uint256) {
138   uint256 c = a + b;
139   if (c < a) return (false, 0);
140   return (true, c);
141   }
142
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 150

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
149   if (b > a) return (false, 0);
150   return (true, a - b);
151   }
152
153   /**
154
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 163

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
162   if (a == 0) return (true, 0);
163   uint256 c = a * b;
164   if (c / a != b) return (false, 0);
165   return (true, c);
166   }
167
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 164

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
163   uint256 c = a * b;
164   if (c / a != b) return (false, 0);
165   return (true, c);
166   }
167
168
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 175

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
174   if (b == 0) return (false, 0);
175   return (true, a / b);
176   }
177
178   /**
179
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 185

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
184   if (b == 0) return (false, 0);
185   return (true, a % b);
186   }
187
188   /**
189
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 199

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
198   function add(uint256 a, uint256 b) internal pure returns (uint256) {
199   uint256 c = a + b;
200   require(c >= a, "SafeMath: addition overflow");
201   return c;
202   }
203
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 216

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
215   require(b <= a, "SafeMath: subtraction overflow");
216   return a - b;
217   }
218
219   /**
220
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 231

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
230   if (a == 0) return 0;
231   uint256 c = a * b;
232   require(c / a == b, "SafeMath: multiplication overflow");
233   return c;
234   }
235
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 232

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
231   uint256 c = a * b;
232   require(c / a == b, "SafeMath: multiplication overflow");
233   return c;
234   }
235
236
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 250

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
249   require(b > 0, "SafeMath: division by zero");
250   return a / b;
251   }
252
253   /**
254
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 267

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
266   require(b > 0, "SafeMath: modulo by zero");
267   return a % b;
268   }
269
270   /**
271
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 285

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
284   require(b <= a, errorMessage);
285   return a - b;
286   }
287
288   /**
289
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 305

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
304   require(b > 0, errorMessage);
305   return a / b;
306   }
307
308   /**
309
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 325

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
324   require(b > 0, errorMessage);
325   return a % b;
326   }
327   }
328
329
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1005

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1004   function tryAdd(uint256 a, uint256 b) internal pure returns (bool, uint256) {
1005   uint256 c = a + b;
1006   if (c < a) return (false, 0);
1007   return (true, c);
1008   }
1009
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1017

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1016   if (b > a) return (false, 0);
1017   return (true, a - b);
1018   }
1019
1020   /**
1021
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1030

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1029   if (a == 0) return (true, 0);
1030   uint256 c = a * b;
1031   if (c / a != b) return (false, 0);
1032   return (true, c);
1033   }
1034
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1031

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1030   uint256 c = a * b;
1031   if (c / a != b) return (false, 0);
1032   return (true, c);
1033   }
1034
1035
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1042

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1041   if (b == 0) return (false, 0);
1042   return (true, a / b);
1043   }
1044
1045   /**
1046
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 1052

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1051   if (b == 0) return (false, 0);
1052   return (true, a % b);
1053   }
1054
1055   /**
1056
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1066

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1065   function add(uint256 a, uint256 b) internal pure returns (uint256) {
1066   uint256 c = a + b;
1067   require(c >= a, "SafeMath: addition overflow");
1068   return c;
1069   }
1070
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1083

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1082   require(b <= a, "SafeMath: subtraction overflow");
1083   return a - b;
1084   }
1085
1086   /**
1087
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1098

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1097   if (a == 0) return 0;
1098   uint256 c = a * b;
1099   require(c / a == b, "SafeMath: multiplication overflow");
1100   return c;
1101   }
1102
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1099

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1098   uint256 c = a * b;
1099   require(c / a == b, "SafeMath: multiplication overflow");
1100   return c;
1101   }
1102
1103
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1117

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1116   require(b > 0, "SafeMath: division by zero");
1117   return a / b;
1118   }
1119
1120   /**
1121
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 1134

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1133   require(b > 0, "SafeMath: modulo by zero");
1134   return a % b;
1135   }
1136
1137   /**
1138
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1152

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1151   require(b <= a, errorMessage);
1152   return a - b;
1153   }
1154
1155   /**
1156
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1172

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1171   require(b > 0, errorMessage);
1172   return a / b;
1173   }
1174
1175   /**
1176
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 1192

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
1191   require(b > 0, errorMessage);
1192   return a % b;
1193   }
1194   }
1195
1196
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 2060

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2059   function mul(int256 a, int256 b) internal pure returns (int256) {
2060   int256 c = a * b;
2061
2062   // Detect overflow when multiplying MIN_INT256 with -1
2063   require(c != MIN_INT256 || (a & MIN_INT256) != (b & MIN_INT256));
2064
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 2064

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2063   require(c != MIN_INT256 || (a & MIN_INT256) != (b & MIN_INT256));
2064   require((b == 0) || (c / b == a));
2065   return c;
2066   }
2067
2068
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 2076

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2075   // Solidity already throws when dividing by 0.
2076   return a / b;
2077   }
2078
2079   /**
2080
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 2083

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2082   function sub(int256 a, int256 b) internal pure returns (int256) {
2083   int256 c = a - b;
2084   require((b >= 0 && c <= a) || (b < 0 && c > a));
2085   return c;
2086   }
2087
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 2092

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2091   function add(int256 a, int256 b) internal pure returns (int256) {
2092   int256 c = a + b;
2093   require((b >= 0 && c >= a) || (b < 0 && c < a));
2094   return c;
2095   }
2096
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 2183

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2182   uint index = map.indexOf[key];
2183   uint lastIndex = map.keys.length - 1;
2184   address lastKey = map.keys[lastIndex];
2185
2186   map.indexOf[lastKey] = index;
2187
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 2314

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2313   //  see https://github.com/ethereum/EIPs/issues/1726#issuecomment-472352728
2314   uint256 internal constant magnitude = 2**128;
2315
2316   uint256 internal magnifiedDividendPerShare;
2317
2318
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 2349

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2348   magnifiedDividendPerShare = magnifiedDividendPerShare.add(
2349   (amount).mul(magnitude) / totalSupply()
2350   );
2351   emit DividendsDistributed(msg.sender, amount);
2352
2353
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 2411

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2410   return
2411   magnifiedDividendPerShare
2412   .mul(balanceOf(_owner))
2413   .toInt256Safe()
2414   .add(magnifiedDividendCorrections[_owner])
2415
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 2564

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2563   require(totalFees <= 100, "Total fee is over 100%");
2564   swapTokensAtAmount = totalSupply_.mul(2).div(10**6); // 0.002%
2565
2566   dividendTracker = BABYTOKENDividendTracker(payable(Clones.clone(implementation)));
2567   dividendTracker.initialize(rewardToken, minimumTokenBalanceForDividends_);
2568
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 2659

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2658   {
2659   for (uint256 i = 0; i < accounts.length; i++) {
2660   _isExcludedFromFees[accounts[i]] = excluded;
2661   }
2662
2663
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 2853

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2852   if (automatedMarketMakerPairs[to]) {
2853   fees += amount.mul(1).div(100);
2854   }
2855   amount = amount.sub(fees);
2856
2857
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 3161

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
3160   while (gasUsed < gas && iterations < numberOfTokenHolders) {
3161   _lastProcessedIndex++;
3162
3163   if (_lastProcessedIndex >= tokenHoldersMap.keys.length) {
3164   _lastProcessedIndex = 0;
3165
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 3171

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
3170   if (processAccount(payable(account), true)) {
3171   claims++;
3172   }
3173   }
3174
3175
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 3175

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
3174
3175   iterations++;
3176
3177   uint256 newGasLeft = gasleft();
3178
3179
```

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

LINE 2183

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2182   uint index = map.indexOf[key];
2183   uint lastIndex = map.keys.length - 1;
2184   address lastKey = map.keys[lastIndex];
2185
2186   map.indexOf[lastKey] = index;
2187
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 2230

low SEVERITY

The current pragma Solidity directive is ""^0.7.6"". 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

- AntiBotBABYTOKEN.sol

Locations

```
2229
2230   pragma solidity ^0.7.6;
2231
2232   // import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
2233   // import "@openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol";
2234
```

SWC-115 | USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.

LINE 2789

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2788   (uint256 iterations, uint256 claims, uint256 lastProcessedIndex) =
dividendTracker.process(gas);
2789   emit ProcessedDividendTracker(iterations, claims, lastProcessedIndex, false, gas,
tx.origin);
2790   }
2791
2792   function claim() external {
2793
```

SWC-115 | USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.

LINE 2873

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2872   ) {
2873   emit ProcessedDividendTracker(iterations, claims, lastProcessedIndex, true, gas,
tx.origin);
2874   } catch {}
2875   }
2876   }
2877
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2154

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2153   function getKeyAtIndex(Map storage map, uint index) public view returns (address)
{
2154   return map.keys[index];
2155   }
2156
2157
2158
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2184

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2183   uint lastIndex = map.keys.length - 1;
2184   address lastKey = map.keys[lastIndex];
2185
2186   map.indexOf[lastKey] = index;
2187   delete map.indexOf[key];
2188
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2189

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2188
2189   map.keys[index] = lastKey;
2190   map.keys.pop();
2191   }
2192   }
2193
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2552

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2551   ) external override initializer {
2552   require(addrs[0] != addrs[3], "Owner and marketing wallet cannot be the same");
2553   __ERC20_init(name_, symbol_);
2554   __Ownable_init();
2555   pinkAntiBot = IPinkAntiBot(addrs[4]);
2556
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2552

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2551   ) external override initializer {
2552   require(addrs[0] != addrs[3], "Owner and marketing wallet cannot be the same");
2553   __ERC20_init(name_, symbol_);
2554   __Ownable_init();
2555   pinkAntiBot = IPinkAntiBot(addrs[4]);
2556
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2555

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2554   __Ownable_init();
2555   pinkAntiBot = IPinkAntiBot(addrs[4]);
2556   pinkAntiBot.setTokenOwner(addrs[0]);
2557   enableAntiBot = true;
2558   rewardToken = addrs[1];
2559
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2556

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2555   pinkAntiBot = IPinkAntiBot(addrs[4]);
2556   pinkAntiBot.setTokenOwner(addrs[0]);
2557   enableAntiBot = true;
2558   rewardToken = addrs[1];
2559   tokenRewardsFee = tokenRewardsFee_;
2560
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2558

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2557   enableAntiBot = true;
2558   rewardToken = addrs[1];
2559   tokenRewardsFee = tokenRewardsFee_;
2560   liquidityFee = liquidityFee_;
2561   marketingFee = marketingFee_;
2562
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2569

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2568
2569   IUniswapV2Router02 _uniswapV2Router = IUniswapV2Router02(addrs[2]);
2570   // Create a uniswap pair for this new token
2571   address _uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(
2572   address(this),
2573
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2579

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2578
2579   _marketingWalletAddress = addrs[3];
2580   // exclude from receiving dividends
2581   dividendTracker.excludeFromDividends(address(dividendTracker));
2582   dividendTracker.excludeFromDividends(address(this));
2583
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2583

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2582   dividendTracker.excludeFromDividends(address(this));
2583   dividendTracker.excludeFromDividends(addrs[0]);
2584   dividendTracker.excludeFromDividends(address(0xdead));
2585   dividendTracker.excludeFromDividends(address(_uniswapV2Router));
2586   // exclude from paying fees or having max transaction amount
2587
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2587

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2586   // exclude from paying fees or having max transaction amount
2587   excludeFromFees(addrs[0], true);
2588   excludeFromFees(_marketingWalletAddress, true);
2589   excludeFromFees(address(this), true);
2590   /*
2591
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2594

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2593   */
2594   _mint(addrs[0], totalSupply_);
2595   transferOwnership(addrs[0]);
2596   }
2597
2598
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2595

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2594   _mint(addrs[0], totalSupply_);
2595   transferOwnership(addrs[0]);
2596   }
2597
2598   function setEnableAntiBot(bool _enable) external onlyOwner {
2599
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2660

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2659   for (uint256 i = 0; i < accounts.length; i++) {
2660   _isExcludedFromFees[accounts[i]] = excluded;
2661   }
2662
2663   emit ExcludeMultipleAccountsFromFees(accounts, excluded);
2664
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2912

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2911   address[] memory path = new address[](2);
2912   path[0] = address(this);
2913   path[1] = uniswapV2Router.WETH();
2914
2915   _approve(address(this), address(uniswapV2Router), tokenAmount);
2916
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2913

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2912   path[0] = address(this);
2913   path[1] = uniswapV2Router.WETH();
2914
2915   _approve(address(this), address(uniswapV2Router), tokenAmount);
2916
2917
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2929

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2928   address[] memory path = new address[](3);
2929   path[0] = address(this);
2930   path[1] = uniswapV2Router.WETH();
2931   path[2] = rewardToken;
2932
2933
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2930

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2929   path[0] = address(this);
2930   path[1] = uniswapV2Router.WETH();
2931   path[2] = rewardToken;
2932
2933   _approve(address(this), address(uniswapV2Router), tokenAmount);
2934
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 2931

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
2930   path[1] = uniswapV2Router.WETH();
2931   path[2] = rewardToken;
2932
2933   _approve(address(this), address(uniswapV2Router), tokenAmount);
2934
2935
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 3167

low SEVERITY

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

Source File

- AntiBotBABYTOKEN.sol

Locations

```
3166
3167   address account = tokenHoldersMap.keys[_lastProcessedIndex];
3168
3169   if (canAutoClaim(lastClaimTimes[account])) {
3170   if (processAccount(payable(account), true)) {
3171
```

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.