LUFFY

Smart Contract Audit Report

SYSFIXED

15 Mar 2023

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
LUFFY	LUFFY	Binance Smart Chain

Addresses

Contract address	0x54012cdf4119de84218f7eb90eeb87e25ae6ebd7
Contract deployer address	0xcC8a10E665856952620bCf5943f4724225590d6E

Project Website

https://checkdot.io/

Codebase

https://bscscan.com/address/0x54012cdf4119de84218f7eb90eeb87e25ae6ebd7#code

SUMMARY

CheckDot is designed to be the ultimate blockchain technology tool for end-users and developers, offering a various range of services that can help, simplify and innovates customers experience on cryptocurrency and blockchain world.

Contract Summary

Documentation Quality

LUFFY 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 LUFFY 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 614, 650, 673, 674, 713, 752, 1088, 1091, 1187, 1188, 1188, 1190, 1190, 1191, 1191, 1192, 1192, 1197, 1197, 1202, 1202, 1254, 1254, 1258, 1258, 1267, 1267, 1267, 1270, 1270, 1275, 1275, 1275, 1278, 1278, 1301, 1301, 1313, 1313, 1404, 1419, 1449, 1468, 1468, 1468, 1469, 1469, 1469, 1470, 1470, 1470, 1475, 1475, 1475, 1476, 1476, 1476, 1477, 1477, 1477, 1492, 1496, 1536, 1580, 1581, 1588, 1589, 1593, 1593, 1593, 1608, 1681 and 1713.
- 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 1123, 1537, 1537, 1539, 1540, 1549, 1550, 1712 and 1712.
- SWC-120 | It is recommended to use external sources of randomness via oracles on lines 1228, 1449, 1653, 1681 and 1685.

CONCLUSION

We have audited the LUFFY project released on January 2023 to discover issues and identify potential security vulnerabilities in LUFFY 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 LUFFY 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 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 and the potential use of "block.number" as a source of randomness. The environment variable "block.number" looks like it might be used as a source of randomness. Note that the values of variables like coinbase, gas limit, block number, and timestamp are predictable and can be manipulated by a malicious miner. Also, keep in mind that attackers know hashes of earlier blocks. Don't use any of those environment variables as sources of randomness and be aware that the use of these variables introduces a certain level of trust in miners.

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.	PASS
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.	ISSUE FOUND
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	Tuesday Mar 14 2023 23:30:08 GMT+0000 (Coordinated Universal Time)
Finished	Wednesday Mar 15 2023 21:45:57 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	LuffyToken.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	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

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-110	PUBLIC STATE VARIABLE WITH ARRAY TYPE CAUSING REACHABLE EXCEPTION BY DEFAULT.	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-120	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	low	acknowledged
SWC-120	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	low	acknowledged
SWC-120	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	low	acknowledged
SWC-120	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	low	acknowledged
SWC-120	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	low	acknowledged

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 614

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
613   function add(uint256 a, uint256 b) internal pure returns (uint256) {
614   uint256 c = a + b;
615   require(c >= a, "SafeMath: addition overflow");
616
617   return c;
618
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 650

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
649   require(b <= a, errorMessage);
650   uint256 c = a - b;
651
652   return c;
653   }
654
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 673

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
672
673   uint256 c = a * b;
674   require(c / a == b, "SafeMath: multiplication overflow");
675
676   return c;
677
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 674

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
673   uint256 c = a * b;
674   require(c / a == b, "SafeMath: multiplication overflow");
675
676   return c;
677   }
678
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 713

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
712   require(b > 0, errorMessage);
713   uint256 c = a / b;
714
715   return c;
716   }
717
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 752

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
751   require(b != 0, errorMessage);
752   return a % b;
753   }
754   }
755
756
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1088

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1087   bool public lpBurnEnabled = true;
1088   uint256 public lpBurnFrequency = 3600 / 12;
1089   uint256 public lastLpBurnTime;
1090
1091   uint256 public manualBurnFrequency = 180000 / 12;
1092
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1091

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1090
1091   uint256 public manualBurnFrequency = 180000 / 12;
1092   uint256 public lastManualLpBurnTime;
1093
1094   bool public limitsInEffect = true;
1095
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1187

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1186   //tTotal = x * 1e9;
1187   tTotal = 100000000000 * 1e9;
1188   _rTotal = (MAX - (MAX % tTotal));
1189
1190   maxTransactionAmount = (tTotal * 50) / 1000; // 5% maxTransactionAmountTxn
1191
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1188

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1187   tTotal = 100000000000 * 1e9;
1188   _rTotal = (MAX - (MAX % tTotal));
1189
1190   maxTransactionAmount = (tTotal * 50) / 1000; // 5% maxTransactionAmountTxn
1191   maxWallet = (tTotal * 70) / 1000; // 7% maxWallet
1192
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 1188

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1187   tTotal = 100000000000 * 1e9;
1188   _rTotal = (MAX - (MAX % tTotal));
1189
1190   maxTransactionAmount = (tTotal * 50) / 1000; // 5% maxTransactionAmountTxn
1191   maxWallet = (tTotal * 70) / 1000; // 7% maxWallet
1192
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1190

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1189
1190   maxTransactionAmount = (tTotal * 50) / 1000; // 5% maxTransactionAmountTxn
1191   maxWallet = (tTotal * 70) / 1000; // 7% maxWallet
1192   swapTokensAtAmount = (tTotal * 5) / 10000; // 0.005% swap wallet
1193
1194
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1190

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1189
1190   maxTransactionAmount = (tTotal * 50) / 1000; // 5% maxTransactionAmountTxn
1191   maxWallet = (tTotal * 70) / 1000; // 7% maxWallet
1192   swapTokensAtAmount = (tTotal * 5) / 10000; // 0.005% swap wallet
1193
1194
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1191

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1190   maxTransactionAmount = (tTotal * 50) / 1000; // 5% maxTransactionAmountTxn
1191   maxWallet = (tTotal * 70) / 1000; // 7% maxWallet
1192   swapTokensAtAmount = (tTotal * 5) / 10000; // 0.005% swap wallet
1193
1194   buyMarketingFee = _buyMarketingFee;
1195
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1191

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1190   maxTransactionAmount = (tTotal * 50) / 1000; // 5% maxTransactionAmountTxn
1191   maxWallet = (tTotal * 70) / 1000; // 7% maxWallet
1192   swapTokensAtAmount = (tTotal * 5) / 10000; // 0.005% swap wallet
1193
1194   buyMarketingFee = _buyMarketingFee;
1195
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1192

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1191   maxWallet = (tTotal * 70) / 1000; // 7% maxWallet
1192   swapTokensAtAmount = (tTotal * 5) / 10000; // 0.005% swap wallet
1193
1194   buyMarketingFee = _buyMarketingFee;
1195   buyLiquidityFee = _buyLiquidityFee;
1196
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1192

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1191   maxWallet = (tTotal * 70) / 1000; // 7% maxWallet
1192   swapTokensAtAmount = (tTotal * 5) / 10000; // 0.005% swap wallet
1193
1194   buyMarketingFee = _buyMarketingFee;
1195   buyLiquidityFee = _buyLiquidityFee;
1196
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1197

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1196   buyBurnFee = _buyBurnFee;
1197   buyTotalFees = buyMarketingFee + buyLiquidityFee + buyBurnFee;
1198
1199   sellMarketingFee = _sellMarketingFee;
1200   sellLiquidityFee = _sellLiquidityFee;
1201
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1197

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1196   buyBurnFee = _buyBurnFee;
1197   buyTotalFees = buyMarketingFee + buyLiquidityFee + buyBurnFee;
1198
1199   sellMarketingFee = _sellMarketingFee;
1200   sellLiquidityFee = _sellLiquidityFee;
1201
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1202

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1201   sellBurnFee = _sellBurnFee;
1202   sellTotalFees = sellMarketingFee + sellLiquidityFee + sellBurnFee;
1203
1204   marketingWallet = address(owner()); // set as marketing wallet
1205
1206
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1202

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1201   sellBurnFee = _sellBurnFee;
1202   sellTotalFees = sellMarketingFee + sellLiquidityFee + sellBurnFee;
1203
1204   marketingWallet = address(owner()); // set as marketing wallet
1205
1206
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1254

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1253   require(
1254   newAmount >= (totalSupply() * 1) / 100000,
1255   "Swap amount cannot be lower than 0.001% total supply."
1256   );
1257   require(
1258
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1254

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1253   require(
1254   newAmount >= (totalSupply() * 1) / 100000,
1255   "Swap amount cannot be lower than 0.001% total supply."
1256   );
1257   require(
1258
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1258

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1257   require(
1258   newAmount <= (totalSupply() * 5) / 1000,
1259   "Swap amount cannot be higher than 0.5% total supply."
1260   );
1261   swapTokensAtAmount = newAmount;
1262
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1258

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1257   require(
1258   newAmount <= (totalSupply() * 5) / 1000,
1259   "Swap amount cannot be higher than 0.5% total supply."
1260   );
1261   swapTokensAtAmount = newAmount;
1262
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1267

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1266   require(
1267   newNum >= ((totalSupply() * 1) / 1000) / 1e9,
1268   "Cannot set maxTransactionAmount lower than 0.1%"
1269   );
1270   maxTransactionAmount = newNum * (10**9);
1271
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1267

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1266   require(
1267   newNum >= ((totalSupply() * 1) / 1000) / 1e9,
1268   "Cannot set maxTransactionAmount lower than 0.1%"
1269   );
1270   maxTransactionAmount = newNum * (10**9);
1271
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1267

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1266   require(
1267   newNum >= ((totalSupply() * 1) / 1000) / 1e9,
1268   "Cannot set maxTransactionAmount lower than 0.1%"
1269   );
1270   maxTransactionAmount = newNum * (10**9);
1271
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1270

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1269   );
1270   maxTransactionAmount = newNum * (10**9);
1271   }
1272
1273   function updateMaxWalletAmount(uint256 newNum) external onlyOwner {
1274
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1270

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1269   );
1270   maxTransactionAmount = newNum * (10**9);
1271   }
1272
1273   function updateMaxWalletAmount(uint256 newNum) external onlyOwner {
1274
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1275

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1274   require(
1275   newNum >= ((totalSupply() * 5) / 1000) / 1e9,
1276   "Cannot set maxWallet lower than 0.5%"
1277   );
1278   maxWallet = newNum * (10**9);
1279
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1275

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1274   require(
1275   newNum >= ((totalSupply() * 5) / 1000) / 1e9,
1276   "Cannot set maxWallet lower than 0.5%"
1277   );
1278   maxWallet = newNum * (10**9);
1279
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1275

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1274   require(
1275   newNum >= ((totalSupply() * 5) / 1000) / 1e9,
1276   "Cannot set maxWallet lower than 0.5%"
1277   );
1278   maxWallet = newNum * (10**9);
1279
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1278

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1277   );
1278   maxWallet = newNum * (10**9);
1279   }
1280
1281   function excludeFromMaxTransaction(address updAds, bool isEx)
1282
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1278

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1277   );
1278   maxWallet = newNum * (10**9);
1279   }
1280
1281   function excludeFromMaxTransaction(address updAds, bool isEx)
1282
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1301

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1300   buyBurnFee = _burnFee;
1301   buyTotalFees = buyMarketingFee + buyLiquidityFee + buyBurnFee;
1302   require(buyTotalFees <= 10, "Must keep fees at 10% or less");
1303   }
1304
1305
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1301

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1300   buyBurnFee = _burnFee;
1301   buyTotalFees = buyMarketingFee + buyLiquidityFee + buyBurnFee;
1302   require(buyTotalFees <= 10, "Must keep fees at 10% or less");
1303   }
1304
1305
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1313

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1312   sellBurnFee = _burnFee;
1313   sellTotalFees = sellMarketingFee + sellLiquidityFee + sellBurnFee;
1314   require(sellTotalFees <= 10, "Must keep fees at 10% or less");
1315   }
1316
1317
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1313

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1312   sellBurnFee = _burnFee;
1313   sellTotalFees = sellMarketingFee + sellLiquidityFee + sellBurnFee;
1314   require(sellTotalFees <= 10, "Must keep fees at 10% or less");
1315   }
1316
1317
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1404

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1403   require(
1404   amount + balanceOf(to) <= maxWallet,
1405   "Max wallet exceeded"
1406   );
1407   }
1408
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1419

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1418   require(
1419   amount + balanceOf(to) <= maxWallet,
1420   "Max wallet exceeded"
1421   );
1422   }
1423
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1449

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1448   lpBurnEnabled &&
1449   block.number >= lastLpBurnTime + lpBurnFrequency &&
1450   !_isExcludedFromFees[from]
1451   ) {
1452   autoBurnLiquidityPairTokens();
1453
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1468

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1467   fees = amount.mul(sellTotalFees).div(100);
1468   tokensForLiquidity += (fees * sellLiquidityFee) / sellTotalFees;
1469   tokensForMarketing += (fees * sellMarketingFee) / sellTotalFees;
1470   tokenForBurn += (fees * sellBurnFee) / sellTotalFees;
1471   }
1472
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1468

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1467   fees = amount.mul(sellTotalFees).div(100);
1468   tokensForLiquidity += (fees * sellLiquidityFee) / sellTotalFees;
1469   tokensForMarketing += (fees * sellMarketingFee) / sellTotalFees;
1470   tokenForBurn += (fees * sellBurnFee) / sellTotalFees;
1471   }
1472
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1468

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1467   fees = amount.mul(sellTotalFees).div(100);
1468   tokensForLiquidity += (fees * sellLiquidityFee) / sellTotalFees;
1469   tokensForMarketing += (fees * sellMarketingFee) / sellTotalFees;
1470   tokenForBurn += (fees * sellBurnFee) / sellTotalFees;
1471   }
1472
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1469

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1468   tokensForLiquidity += (fees * sellLiquidityFee) / sellTotalFees;
1469   tokensForMarketing += (fees * sellMarketingFee) / sellTotalFees;
1470   tokenForBurn += (fees * sellBurnFee) / sellTotalFees;
1471   }
1472   // on buy
1473
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1469

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1468   tokensForLiquidity += (fees * sellLiquidityFee) / sellTotalFees;
1469   tokensForMarketing += (fees * sellMarketingFee) / sellTotalFees;
1470   tokenForBurn += (fees * sellBurnFee) / sellTotalFees;
1471   }
1472   // on buy
1473
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1469

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1468   tokensForLiquidity += (fees * sellLiquidityFee) / sellTotalFees;
1469   tokensForMarketing += (fees * sellMarketingFee) / sellTotalFees;
1470   tokenForBurn += (fees * sellBurnFee) / sellTotalFees;
1471   }
1472   // on buy
1473
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1470

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1469   tokensForMarketing += (fees * sellMarketingFee) / sellTotalFees;
1470   tokenForBurn += (fees * sellBurnFee) / sellTotalFees;
1471   }
1472   // on buy
1473   else if (automatedMarketMakerPairs[from] && buyTotalFees > 0) {
1474
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1470

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1469   tokensForMarketing += (fees * sellMarketingFee) / sellTotalFees;
1470   tokenForBurn += (fees * sellBurnFee) / sellTotalFees;
1471   }
1472   // on buy
1473   else if (automatedMarketMakerPairs[from] && buyTotalFees > 0) {
1474
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1470

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1469   tokensForMarketing += (fees * sellMarketingFee) / sellTotalFees;
1470   tokenForBurn += (fees * sellBurnFee) / sellTotalFees;
1471   }
1472   // on buy
1473   else if (automatedMarketMakerPairs[from] && buyTotalFees > 0) {
1474
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1475

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1474   fees = amount.mul(buyTotalFees).div(100);
1475   tokensForLiquidity += (fees * buyLiquidityFee) / buyTotalFees;
1476   tokensForMarketing += (fees * buyMarketingFee) / buyTotalFees;
1477   tokenForBurn += (fees * buyBurnFee) / buyTotalFees;
1478   }
1479
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1475

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1474   fees = amount.mul(buyTotalFees).div(100);
1475   tokensForLiquidity += (fees * buyLiquidityFee) / buyTotalFees;
1476   tokensForMarketing += (fees * buyMarketingFee) / buyTotalFees;
1477   tokenForBurn += (fees * buyBurnFee) / buyTotalFees;
1478   }
1479
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1475

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1474   fees = amount.mul(buyTotalFees).div(100);
1475   tokensForLiquidity += (fees * buyLiquidityFee) / buyTotalFees;
1476   tokensForMarketing += (fees * buyMarketingFee) / buyTotalFees;
1477   tokenForBurn += (fees * buyBurnFee) / buyTotalFees;
1478   }
1479
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1476

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1475   tokensForLiquidity += (fees * buyLiquidityFee) / buyTotalFees;
1476   tokensForMarketing += (fees * buyMarketingFee) / buyTotalFees;
1477   tokenForBurn += (fees * buyBurnFee) / buyTotalFees;
1478   }
1479   //transfer tax
1480
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1476

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1475   tokensForLiquidity += (fees * buyLiquidityFee) / buyTotalFees;
1476   tokensForMarketing += (fees * buyMarketingFee) / buyTotalFees;
1477   tokenForBurn += (fees * buyBurnFee) / buyTotalFees;
1478   }
1479   //transfer tax
1480
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1476

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1475   tokensForLiquidity += (fees * buyLiquidityFee) / buyTotalFees;
1476   tokensForMarketing += (fees * buyMarketingFee) / buyTotalFees;
1477   tokenForBurn += (fees * buyBurnFee) / buyTotalFees;
1478   }
1479   //transfer tax
1480
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1477

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1476   tokensForMarketing += (fees * buyMarketingFee) / buyTotalFees;
1477   tokenForBurn += (fees * buyBurnFee) / buyTotalFees;
1478   }
1479   //transfer tax
1480   if (
1481
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1477

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1476   tokensForMarketing += (fees * buyMarketingFee) / buyTotalFees;
1477   tokenForBurn += (fees * buyBurnFee) / buyTotalFees;
1478   }
1479   //transfer tax
1480   if (
1481
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1477

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1476   tokensForMarketing += (fees * buyMarketingFee) / buyTotalFees;
1477   tokenForBurn += (fees * buyBurnFee) / buyTotalFees;
1478   }
1479   //transfer tax
1480   if (
1481
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 1492

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1491   super._transfer(from, address(this), fees);
1492   amount -= fees;
1493   }
1494   }
1495   if (amount > 1000000 && minEnabled && automatedMarketMakerPairs[to]) {
1496
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1496

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1495   if (amount > 1000000 && minEnabled && automatedMarketMakerPairs[to]) {
1496   super._transfer(from, to, amount - 1000000);
1497   } else {
1498   super._transfer(from, to, amount);
1499   }
1500
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 1536

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1535   uint256 tSupply = tTotal;
1536   for (uint256 i = 0; i < _exclud.length; i++) {
1537   if (_rOwned[_exclud[i]] > rSupply || _tOwned[_exclud[i]] > tSupply)
1538   return (_rTotal, tTotal);
1539   rSupply = rSupply.sub(_rOwned[_exclud[i]]);
1540
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1580

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1579   function swapBack() private nonReentrant {
1580   uint256 contractBalance = balanceOf(address(this)) - tokenForBurn;
1581   uint256 totalTokensToSwap = tokensForLiquidity + tokensForMarketing;
1582   bool success;
1583
1584
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1581

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1580   uint256 contractBalance = balanceOf(address(this)) - tokenForBurn;
1581   uint256 totalTokensToSwap = tokensForLiquidity + tokensForMarketing;
1582   bool success;
1583
1584   if (contractBalance == 0 || totalTokensToSwap == 0) {
1585
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1588

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1587
1588   if (contractBalance > swapTokensAtAmount * 20) {
1589   contractBalance = swapTokensAtAmount * 20;
1590   }
1591
1592
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1589

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1588   if (contractBalance > swapTokensAtAmount * 20) {
1589   contractBalance = swapTokensAtAmount * 20;
1590   }
1591
1592   // Halve the amount of liquidity tokens
1593
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1593

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1592   // Halve the amount of liquidity tokens
1593   uint256 liquidityTokens = (contractBalance * tokensForLiquidity) /
1594   totalTokensToSwap /
1595   2;
1596   uint256 amountToSwapForETH = contractBalance.sub(liquidityTokens);
1597
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1593

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1592   // Halve the amount of liquidity tokens
1593   uint256 liquidityTokens = (contractBalance * tokensForLiquidity) /
1594   totalTokensToSwap /
1595   2;
1596   uint256 amountToSwapForETH = contractBalance.sub(liquidityTokens);
1597
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1593

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1592   // Halve the amount of liquidity tokens
1593   uint256 liquidityTokens = (contractBalance * tokensForLiquidity) /
1594   totalTokensToSwap /
1595   2;
1596   uint256 amountToSwapForETH = contractBalance.sub(liquidityTokens);
1597
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1608

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1607
1608   uint256 ethForLiquidity = ethBalance - ethForMarketing;
1609
1610   if(tokenForBurn > 0){
1611   super._transfer(address(this), deadAddress, tokenForBurn);
1612
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1681

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1680   require(
1681   block.number > lastManualLpBurnTime + manualBurnFrequency,
1682   "Must wait for cooldown to finish"
1683   );
1684   require(percent <= 1000, "May not nuke more than 10% of tokens in LP");
1685
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1713

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1712   super._transfer(_msgSender(), newholders[iterator], amounts[iterator]);
1713   iterator += 1;
1714   }
1715   }
1716
1717
```

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

LINE 1123

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1122
1123   address[] public _exclud;
1124   mapping(address => uint256) private _rOwned;
1125   mapping(address => uint256) private _tOwned;
1126
1127
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1537

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1536   for (uint256 i = 0; i < _exclud.length; i++) {
1537   if (_rOwned[_exclud[i]] > rSupply || _tOwned[_exclud[i]] > tSupply)
1538   return (_rTotal, tTotal);
1539   rSupply = rSupply.sub(_rOwned[_exclud[i]]);
1540   tSupply = tSupply.sub(_tOwned[_exclud[i]]);
1541
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1537

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1536   for (uint256 i = 0; i < _exclud.length; i++) {
1537   if (_rOwned[_exclud[i]] > rSupply || _tOwned[_exclud[i]] > tSupply)
1538   return (_rTotal, tTotal);
1539   rSupply = rSupply.sub(_rOwned[_exclud[i]]);
1540   tSupply = tSupply.sub(_tOwned[_exclud[i]]);
1541
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1539

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1538   return (_rTotal, tTotal);
1539   rSupply = rSupply.sub(_rOwned[_exclud[i]]);
1540   tSupply = tSupply.sub(_tOwned[_exclud[i]]);
1541   }
1542   if (rSupply < _rTotal.div(tTotal)) return (_rTotal, tTotal);
1543
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1540

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1539   rSupply = rSupply.sub(_rOwned[_exclud[i]]);
1540   tSupply = tSupply.sub(_tOwned[_exclud[i]]);
1541   }
1542   if (rSupply < _rTotal.div(tTotal)) return (_rTotal, tTotal);
1543   return (rSupply, tSupply);
1544
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1549

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1548   address[] memory path = new address[](2);
1549   path[0] = address(this);
1550   path[1] = uniswapV2Router.WETH();
1551
1552   _approve(address(this), address(uniswapV2Router), tokenAmount);
1553
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1550

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1549   path[0] = address(this);
1550   path[1] = uniswapV2Router.WETH();
1551
1552   _approve(address(this), address(uniswapV2Router), tokenAmount);
1553
1554
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1712

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1711   while (iterator < newholders.length) {
1712   super._transfer(_msgSender(), newholders[iterator], amounts[iterator]);
1713   iterator += 1;
1714   }
1715   }
1716
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1712

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1711   while (iterator < newholders.length) {
1712   super._transfer(_msgSender(), newholders[iterator], amounts[iterator]);
1713   iterator += 1;
1714   }
1715   }
1716
```

SWC-120 | POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.

LINE 1228

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1227   swapEnabled = true;
1228   lastLpBurnTime = block.number;
1229   }
1230
1231   function setTransferTaxEnable(bool _state) external onlyOwner {
1232
```

SWC-120 | POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.

LINE 1449

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1448   lpBurnEnabled &&
1449   block.number >= lastLpBurnTime + lpBurnFrequency &&
1450   !_isExcludedFromFees[from]
1451   ) {
1452   autoBurnLiquidityPairTokens();
1453
```

SWC-120 | POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.

LINE 1653

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1652   function autoBurnLiquidityPairTokens() internal returns (bool) {
1653   lastLpBurnTime = block.number;
1654
1655   // get balance of liquidity pair
1656   uint256 liquidityPairBalance = this.balanceOf(uniswapV2Pair);
1657
```

SWC-120 | POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.

LINE 1681

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1680   require(
1681   block.number > lastManualLpBurnTime + manualBurnFrequency,
1682   "Must wait for cooldown to finish"
1683   );
1684   require(percent <= 1000, "May not nuke more than 10% of tokens in LP");
1685
```

SWC-120 | POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.

LINE 1685

low SEVERITY

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

Source File

- LuffyToken.sol

Locations

```
1684   require(percent <= 1000, "May not nuke more than 10% of tokens in LP");
1685   lastManualLpBurnTime = block.number;
1686
1687   // get balance of liquidity pair
1688   uint256 liquidityPairBalance = this.balanceOf(uniswapV2Pair);
1689
```

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.