BANANA BANANA

Smart Contract Audit Report

SYSFIXED

12 Dec 2022

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
BANANA BANANA	BBFT	BSC

Addresses

Contract address	0x5D7511c14f908B99f5bB000143bDa13759b55C48
Contract deployer address	0xE3eFc5AD5CbcB7214Ac5AB92FE7344344191f246

Project Website

https://www.bananabanana.fun/

Codebase

https://bscscan.com/address/0x5D7511c14f908B99f5bB000143bDa13759b55C48#code

SUMMARY

Banana banana project is new way to reshape fun creativity to money, yes to nft ,web3 , staking dapp, fun creativity app nft marketplace for fun trading & nft, earn and trade and upvote. The benefit is trending, staking dapp 1% daily earn & pool rewards system, community competition web3 dapp, new level of the social fun network to the millions user, bbft primary utilities token fabricated on multi-fun project, team partner, bbft lifetime opportunity and will touch the zenith.

Contract Summary

Documentation Quality

BANANA BANANA provides a document with a very good standard of solidity base code.

- The technical description is provided clearly and structured also don't have any risk issue.

Code Quality

The Overall quality of the basecode is GOOD

- Standart solidity basecode and rules are already followed with Coinhound Project .

Test Coverage

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

Audit Findings Summary

- SWC-101 | Arithmetic operation Issues discovered on lines 169, 170, 172, 173, 174, 175, 311, 323, 337, 390, 405, 407, 425, 426, 430, 433, 435, 439, 442, 444, 480, 481, 482, 483, 503, 509, 510, 511, 513, 519, 525, 530, 531, 533, 565, 579, 584, 621, 632, 636, 639, 640, 645, 656, 657, 657, 659, 665, 666, 667, 674, 718, 724, 739, 745, and 407.
- SWC-103 | A floating pragma is set on lines 6. The current pragma Solidity directive is ""^0.8.17"". 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.
- SWC-110 | Out of bounds array access on lines 406, 407, 527, 528, 530, 531, 698, 699, and 740.
- SWC-120 | OPotential use of "block.number" as source of randonmness on lines 177 and 621.

CONCLUSION

CONCLUSION

We have audited the Goge Coin which has been released to discover issues and identify potential security vulnerabilities in Goge Project. This process is used to find bugs, technical issues, and security loopholes that find some common issues in the code.

The security audit report produced satisfactory results with a low risk issue on the contract project.

The most common issue found in writing code on contracts that do not pose a big risk, writing on contracts is close to the standard of writing contracts in general. Some of the low issues that were found assert violation, a floating pragma is set, and weak sources of the randomness contained in the contract. We recommend to don't using any of those environment variables as sources of randomness and being aware that the use of these variables introduces a certain level of trust into 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.	ISSUE FOUND
Unchecked Call Return Value	SWC-104	The return value of a message call should be checked.	PASS
SELFDESTRUCT Instruction	SWC-106	The contract should not be self-destructible while it has funds belonging to users.	PASS
Check-Effect Interaction	SWC-107	Check-Effect-Interaction pattern should be followed if the code performs ANY external call.	PASS
Assert Violation	SWC-110	Properly functioning code should never reach a failing assert statement.	ISSUE FOUND
Deprecated Solidity Functions	SWC-111	Deprecated built-in functions should never be used.	PASS
Delegate call to Untrusted Caller	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
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
Incorrect Inheritance Order	SWC-125	When inheriting multiple contracts, especially if they have identical functions, a developer should carefully specify inheritance in the correct order. The rule of thumb is to inherit contracts from more /general/ to more /specific/.	PASS

SMART CONTRACT ANALYSIS

Started	Sun Dec 11 2022 02:40:46 GMT+0000 (Coordinated Universal Time)
Finished	Mon Dec 12 2022 03:41:41 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	Bananabanana.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	COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-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 169

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
168
169   uint256 private _tTotal = 100000000 * 10**_decimals;
170   uint256 private _rTotal = (MAX - (MAX % _tTotal));
171
172   uint256 public swapTokensAtAmount = 1_000_000 * 10**_decimals;
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 170

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
169   uint256 private _tTotal = 100000000 * 10**_decimals;
170   uint256 private _rTotal = (MAX - (MAX % _tTotal));
171
172   uint256 public swapTokensAtAmount = 1_000_000 * 10**_decimals;
173   uint256 public maxBuyLimit = 1_000_000 * 10**_decimals;
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 172

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
171
172   uint256 public swapTokensAtAmount = 1_000_000 * 10**_decimals;
173   uint256 public maxBuyLimit = 1_000_000 * 10**_decimals;
174   uint256 public maxSellLimit = 1_000_000 * 10**_decimals;
175   uint256 public maxWalletLimit = 100_000_000 * 10**_decimals;
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 173

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
172   uint256 public swapTokensAtAmount = 1_000_000 * 10**_decimals;
173   uint256 public maxBuyLimit = 1_000_000 * 10**_decimals;
174   uint256 public maxSellLimit = 1_000_000 * 10**_decimals;
175   uint256 public maxWalletLimit = 100_000_000 * 10**_decimals;
176
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 174

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
173   uint256 public maxBuyLimit = 1_000_000 * 10**_decimals;
174   uint256 public maxSellLimit = 1_000_000 * 10**_decimals;
175   uint256 public maxWalletLimit = 100_000_000 * 10**_decimals;
176
177   uint256 public genesis_block = block.number;
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 175

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
174   uint256 public maxSellLimit = 1_000_000 * 10**_decimals;
175   uint256 public maxWalletLimit = 100_000_000 * 10**_decimals;
176
177   uint256 public genesis_block = block.number;
178   uint256 private deadline;
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 311

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
310   );
311   _approve(sender, _msgSender(), currentAllowance - amount);
312
313   return true;
314   }
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 323

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
322   spender,
323   _allowances[_msgSender()][spender] + addedValue
324   );
325   return true;
326   }
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 337

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
336   );
337   _approve(_msgSender(), spender, currentAllowance - subtractedValue);
338
339   return true;
340   }
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 390

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
389   uint256 currentRate = _getRate();
390   return rAmount / currentRate;
391   }
392
393   //@dev kept original RFI naming -> "reward" as in reflection
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 405

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
404   require(_isExcluded[account], "Account is not excluded");
405   for (uint256 i = 0; i < _excluded.length; i++) {
406   if (_excluded[i] == account) {
407   _excluded[i] = _excluded[_excluded.length - 1];
408   _tOwned[account] = 0;
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 407

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
406   if (_excluded[i] == account) {
407   _excluded[i] = _excluded[_excluded.length - 1];
408   _tOwned[account] = 0;
409   _isExcluded[account] = false;
410   _excluded.pop();
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 425

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
424   function _reflectRfi(uint256 rRfi, uint256 tRfi) private {
425   _rTotal -= rRfi;
426   totFeesPaid.rfi += tRfi;
427   }
428
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 426

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
425   _rTotal -= rRfi;
426   totFeesPaid.rfi += tRfi;
427   }
428
429   function _takeLiquidity(uint256 rLiquidity, uint256 tLiquidity) private {
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 430

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
429   function _takeLiquidity(uint256 rLiquidity, uint256 tLiquidity) private {
430   totFeesPaid.liquidity += tLiquidity;
431
432   if (_isExcluded[address(this)]) {
433   _tOwned[address(this)] += tLiquidity;
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 433

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
432   if (_isExcluded[address(this)]) {
433   _tOwned[address(this)] += tLiquidity;
434   }
435   _rOwned[address(this)] += rLiquidity;
436   }
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 435

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
434   }
435   _rOwned[address(this)] += rLiquidity;
436   }
437
438   function _takeMarketing(uint256 rMarketing, uint256 tMarketing) private {
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 439

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
438   function _takeMarketing(uint256 rMarketing, uint256 tMarketing) private {
439   totFeesPaid.marketing += tMarketing;
440
441   if (_isExcluded[address(this)]) {
442   _tOwned[address(this)] += tMarketing;
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 442

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
441   if (_isExcluded[address(this)]) {
442   _tOwned[address(this)] += tMarketing;
443   }
444   _rOwned[address(this)] += rMarketing;
445   }
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 444

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
443   }
444   _rOwned[address(this)] += rMarketing;
445   }
446
447   function _getValues(
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 480

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
479
480   s.tRfi = (tAmount * temp.rfi) / 100;
481   s.tMarketing = (tAmount * temp.marketing) / 100;
482   s.tLiquidity = (tAmount * temp.liquidity) / 100;
483   s.tTransferAmount = tAmount - s.tRfi - s.tMarketing - s.tLiquidity;
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 481

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
480   s.tRfi = (tAmount * temp.rfi) / 100;
481   s.tMarketing = (tAmount * temp.marketing) / 100;
482   s.tLiquidity = (tAmount * temp.liquidity) / 100;
483   s.tTransferAmount = tAmount - s.tRfi - s.tMarketing - s.tLiquidity;
484   return s;
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 482

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
481   s.tMarketing = (tAmount * temp.marketing) / 100;
482   s.tLiquidity = (tAmount * temp.liquidity) / 100;
483   s.tTransferAmount = tAmount - s.tRfi - s.tMarketing - s.tLiquidity;
484   return s;
485   }
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 483

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
482   s.tLiquidity = (tAmount * temp.liquidity) / 100;
483   s.tTransferAmount = tAmount - s.tRfi - s.tMarketing - s.tLiquidity;
484   return s;
485   }
486
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 503

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
502   {
503   rAmount = tAmount * currentRate;
504
505   if (!takeFee) {
506   return (rAmount, rAmount, 0, 0, 0);
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 509

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
508
509   rRfi = s.tRfi * currentRate;
510   rMarketing = s.tMarketing * currentRate;
511   rLiquidity = s.tLiquidity * currentRate;
512
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 510

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
509   rRfi = s.tRfi * currentRate;
510   rMarketing = s.tMarketing * currentRate;
511   rLiquidity = s.tLiquidity * currentRate;
512
513   rTransferAmount = rAmount - rRfi - rMarketing - rLiquidity;
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 511

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
510   rMarketing = s.tMarketing * currentRate;
511   rLiquidity = s.tLiquidity * currentRate;
512
513   rTransferAmount = rAmount - rRfi - rMarketing - rLiquidity;
514   return (rAmount, rTransferAmount, rRfi, rMarketing, rLiquidity);
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 513

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
512
513   rTransferAmount = rAmount - rRfi - rMarketing - rLiquidity;
514   return (rAmount, rTransferAmount, rRfi, rMarketing, rLiquidity);
515   }
516
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 519

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
518   (uint256 rSupply, uint256 tSupply) = _getCurrentSupply();
519   return rSupply / tSupply;
520   }
521
522   function _getCurrentSupply() private view returns (uint256, uint256) {
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 525

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
524   uint256 tSupply = _tTotal;
525   for (uint256 i = 0; i < _excluded.length; i++) {
526   if (
527   _rOwned[_excluded[i]] > rSupply ||
528   _tOwned[_excluded[i]] > tSupply
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 530

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
529   ) return (_rTotal, _tTotal);
530   rSupply = rSupply - _rOwned[_excluded[i]];
531   tSupply = tSupply - _tOwned[_excluded[i]];
532   }
533   if (rSupply < _rTotal / _tTotal) return (_rTotal, _tTotal);
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 531

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
530   rSupply = rSupply - _rOwned[_excluded[i]];
531   tSupply = tSupply - _tOwned[_excluded[i]];
532   }
533   if (rSupply < _rTotal / _tTotal) return (_rTotal, _tTotal);
534   return (rSupply, tSupply);
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 533

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
532   }
533   if (rSupply < _rTotal / _tTotal) return (_rTotal, _tTotal);
534   return (rSupply, tSupply);
535   }
536
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 565

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
564   require(
565   balanceOf(to) + amount <= maxWalletLimit,
566   "You are exceeding maxWalletLimit"
567   );
568   }
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 579

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
578   require(
579   balanceOf(to) + amount <= maxWalletLimit,
580   "You are exceeding maxWalletLimit"
581   );
582   }
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 584

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
583   if (coolDownEnabled) {
584   uint256 timePassed = block.timestamp - _lastSell[from];
585   require(timePassed >= coolDownTime, "Cooldown enabled");
586   _lastSell[from] = block.timestamp;
587   }
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 621

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
620   !_isExcludedFromFee[recipient] &&
621   block.number <= genesis_block + deadline;
622
623   valuesFromGetValues memory s = _getValues(
624   tAmount,
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 632

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
631   //from excluded
632   _tOwned[sender] = _tOwned[sender] - tAmount;
633   }
634   if (_isExcluded[recipient]) {
635   //to excluded
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 636

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
635   //to excluded
636   _tOwned[recipient] = _tOwned[recipient] + s.tTransferAmount;
637   }
638
639   _rOwned[sender] = _rOwned[sender] - s.rAmount;
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 640

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
639   _rOwned[sender] = _rOwned[sender] - s.rAmount;
640   _rOwned[recipient] = _rOwned[recipient] + s.rTransferAmount;
641
642   if (s.rRfi > 0 || s.tRfi > 0) _reflectRfi(s.rRfi, s.tRfi);
643   if (s.rLiquidity > 0 || s.tLiquidity > 0) {
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 645

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
644   _takeLiquidity(s.rLiquidity, s.tLiquidity);
645   emit Transfer(sender, address(this), s.tLiquidity + s.tMarketing);
646   }
647   if (s.rMarketing > 0 || s.tMarketing > 0)
648   _takeMarketing(s.rMarketing, s.tMarketing);
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 656

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
655   {
656   uint256 denominator = (temp.liquidity + temp.marketing) * 2;
657   uint256 tokensToAddLiquidityWith = (contractBalance * temp.liquidity) /
658   denominator;
659   uint256 toSwap = contractBalance - tokensToAddLiquidityWith;
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 657

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
uint256 denominator = (temp.liquidity + temp.marketing) * 2;
uint256 tokensToAddLiquidityWith = (contractBalance * temp.liquidity) /
denominator;
uint256 toSwap = contractBalance - tokensToAddLiquidityWith;
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 657

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
656   uint256 denominator = (temp.liquidity + temp.marketing) * 2;
657   uint256 tokensToAddLiquidityWith = (contractBalance * temp.liquidity) /
658   denominator;
659   uint256 toSwap = contractBalance - tokensToAddLiquidityWith;
660
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 659

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
658   denominator;
659   uint256 toSwap = contractBalance - tokensToAddLiquidityWith;
660
661   uint256 initialBalance = address(this).balance;
662
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 665

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
664
665   uint256 deltaBalance = address(this).balance - initialBalance;
666   uint256 unitBalance = deltaBalance / (denominator - temp.liquidity);
667   uint256 bnbToAddLiquidityWith = unitBalance * temp.liquidity;
668
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 666

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
665   uint256 deltaBalance = address(this).balance - initialBalance;
666   uint256 unitBalance = deltaBalance / (denominator - temp.liquidity);
667   uint256 bnbToAddLiquidityWith = unitBalance * temp.liquidity;
668
669   if (bnbToAddLiquidityWith > 0) {
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 667

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
666   uint256 unitBalance = deltaBalance / (denominator - temp.liquidity);
667   uint256 bnbToAddLiquidityWith = unitBalance * temp.liquidity;
668
669   if (bnbToAddLiquidityWith > 0) {
670   // Add liquidity to pancake
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 674

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
673
674   uint256 marketingAmt = unitBalance * 2 * temp.marketing;
675   if (marketingAmt > 0) {
676   payable(marketingWallet).sendValue(marketingAmt);
677   }
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 718

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
717   function updateCooldown(bool state, uint8 time) external onlyOwner {
718   coolDownTime = time * 1 seconds;
719   coolDownEnabled = state;
720   }
721
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 724

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
if(amount > 0){
swapTokensAtAmount = amount * 10**_decimals;
}
}

```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 739

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
738   {
739   for (uint256 i = 0; i < accounts.length; i++) {
740   allowedTransfer[accounts[i]] = state;
741   }
742   }
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 745

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
744   function updateMaxWalletlimit(uint256 amount) external onlyOwner {
745   maxWalletLimit = amount * 10**decimals();
746   }
747
748   function updateRouterAndPair(address routerAddress )
```

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

LINE 407

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
406   if (_excluded[i] == account) {
407   _excluded[i] = _excluded[_excluded.length - 1];
408   _tOwned[account] = 0;
409   _isExcluded[account] = false;
410   _excluded.pop();
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 6

low SEVERITY

The current pragma Solidity directive is ""^0.8.7"". 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

- Bananabanana.sol

Locations

```
5   // SPDX-License-Identifier: UNLICENSE
6   pragma solidity ^0.8.7;
7
8   interface IERC20 {
9   function totalSupply() external view returns (uint256);
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 406

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
405   for (uint256 i = 0; i < _excluded.length; i++) {
406   if (_excluded[i] == account) {
407   _excluded[i] = _excluded[_excluded.length - 1];
408   _tOwned[account] = 0;
409   _isExcluded[account] = false;
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 407

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
406   if (_excluded[i] == account) {
407   _excluded[i] = _excluded[_excluded.length - 1];
408   _tOwned[account] = 0;
409   _isExcluded[account] = false;
410   _excluded.pop();
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 527

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
526   if (
527   _rOwned[_excluded[i]] > rSupply ||
528   _tOwned[_excluded[i]] > tSupply
529   ) return (_rTotal, _tTotal);
530   rSupply = rSupply - _rOwned[_excluded[i]];
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 528

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
527   _rOwned[_excluded[i]] > rSupply ||
528   _tOwned[_excluded[i]] > tSupply
529   ) return (_rTotal, _tTotal);
530   rSupply = rSupply - _rOwned[_excluded[i]];
531   tSupply = tSupply - _tOwned[_excluded[i]];
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 530

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
529   ) return (_rTotal, _tTotal);
530   rSupply = rSupply - _rOwned[_excluded[i]];
531   tSupply = tSupply - _tOwned[_excluded[i]];
532   }
533   if (rSupply < _rTotal / _tTotal) return (_rTotal, _tTotal);
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 531

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
530   rSupply = rSupply - _rOwned[_excluded[i]];
531   tSupply = tSupply - _tOwned[_excluded[i]];
532   }
533   if (rSupply < _rTotal / _tTotal) return (_rTotal, _tTotal);
534   return (rSupply, tSupply);
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 698

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
697   address[] memory path = new address[](2);
698   path[0] = address(this);
699   path[1] = router.WETH();
700
701   _approve(address(this), address(router), tokenAmount);
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 699

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
698   path[0] = address(this);
699   path[1] = router.WETH();
700
701   _approve(address(this), address(router), tokenAmount);
702
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 740

low SEVERITY

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

Source File

- Bananabanana.sol

Locations

```
739   for (uint256 i = 0; i < accounts.length; i++) {
740   allowedTransfer[accounts[i]] = state;
741   }
742   }
743
```

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

LINE 177

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

- Bananabanana.sol

Locations

```
176   uint256 public genesis_block = block.number;
177   uint256 private deadline;
178
179   address public deadWallet = 0x000000000000000000000000000000000000dEaD;
```

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

LINE 621

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

- Bananabanana.sol

Locations

```
620   !_isExcludedFromFee[recipient] &&
621   block.number <= genesis_block + deadline;
622
623   valuesFromGetValues memory s = _getValues(
624   tAmount,
```

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.