Lunar Rabbit

Smart Contract Audit Report

SYSFIXED

11 Jan 2023

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
Lunar Rabbit	LunarRabbit	Binance Smart Chain

Addresses

Contract address	0xCF136913c4583aD8D12190DeA731e6FA75F45E95
Contract deployer address	0x9B93a0400e163b6d380D6CC1Ef98D13175B3B357

Project Website

https://www.lunarrabbit.cz/

Codebase

https://bscscan.com/address/0xCF136913c4583aD8D12190DeA731e6FA75F45E95#contracts

SUMMARY

Dev team from project x70 - the lowest project is x6 - no project yet rug and never, all still run, Nft + stake + game: Ready to launch, Cmc & Cgk list in 12 hrs - SURE there will definitely be an x1000 project

Contract Summary

Documentation Quality

Lunar Rabbit 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.

- Standart solidity basecode and rules are already followed with Lunar Rabbit 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 519, 552, 568, 570, 604, 650, 711, 715, 730, 738, 750, 990, 991, 997, 1000, 1004, 1088, 1097, 1098, 1099, 1103, 1105, 11061127, 1138, 1150, 1222, 1239, 1260, 1261,1262, 1263, 1264, 1269, 1270, 1275, 1278, 1280, 1281, 1285, 1287, 1288, 1290, 1291, 1293, 1294, 1300,1344, 1345, 1346, 1347, 1356, 1363, 1366, 1383 and 1383.
- SWC-103 | Pragma statements can be allowed to float when a contract is intended on lines 47.
- SWC-110 | It is recommended to use use of revert(), assert(), and require() in Solidity, and the new REVERT opcode in the EVM on lines 1088, 1092, 1307 and 1308.

CONCLUSION

We have audited the Lunar Rabbit Coin which has released on January 2023 to discover issues and identify potential security vulnerabilities in Lunar Rabbit 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 stated variable visibility are not set, and a floating pragma is set. We recommended specifying 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.

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
Reentrancy	SWC-107	Check effect interaction pattern should be followed if the code performs recursive 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.	PASS
Incorrect Inheritance Order	SWC-125	When inheriting multiple contracts, especially if they have identical functions, a developer should carefully specify inheritance in the correct order. The rule of thumb is to inherit contracts from more /general/ to more /specific/.	PASS

SMART CONTRACT ANALYSIS

Started	Tuesday Jan 10 2023 01:12:29 GMT+0000 (Coordinated Universal Time)
Finished	Wednesday Jan 11 2023 11:42:47 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	LunarRabbit.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-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-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 519

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
518   */
519   function sub(uint256 a, uint256 b, string memory errorMessage) internal pure
returns (uint256) {
520   require(b <= a, errorMessage);
521   uint256 c = a - b;
522
523
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 552

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
551   * @dev Returns the integer division of two unsigned integers. Reverts on
552   * division by zero. The result is rounded towards zero.
553   *
554   * Counterpart to Solidity's `/` operator. Note: this function uses a
555   * `revert` opcode (which leaves remaining gas untouched) while Solidity
556
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 568

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
567   * @dev Returns the integer division of two unsigned integers. Reverts with custom
message on
568   * division by zero. The result is rounded towards zero.
569   *
570   * Counterpart to Solidity's `/` operator. Note: this function uses a
571   * `revert` opcode (which leaves remaining gas untouched) while Solidity
572
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 570

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
569   *
570   * Counterpart to Solidity's `/` operator. Note: this function uses a
571   * `revert` opcode (which leaves remaining gas untouched) while Solidity
572   * uses an invalid opcode to revert (consuming all remaining gas).
573   *
574
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 604

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
603   * @dev Returns the remainder of dividing two unsigned integers. (unsigned integer
modulo),
604   * Reverts with custom message when dividing by zero.
605   *
606   * Counterpart to Solidity's `%` operator. This function uses a `revert`
607   * opcode (which leaves remaining gas untouched) while Solidity uses an
608
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 650

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
649   /**
650   * @dev Leaves the contract without owner. It will not be possible to call
651   * `onlyOwner` functions anymore. Can only be called by the current owner.
652   *
653   * NOTE: Renouncing ownership will leave the contract without an owner,
654
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 711

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
710   /**
711   * @dev Adds two int256 variables and fails on overflow.
712   */
713   function add(int256 a, int256 b) internal pure returns (int256) {
714   int256 c = a + b;
715
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 715

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
714   int256 c = a + b;
715   require((b >= 0 && c >= a) || (b < 0 && c < a));
716   return c;
717   }
718
719
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 730

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
729   require(a >= 0);
730   return uint256(a);
731   }
732   }
733
734
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 738

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
737   require(b >= 0);
738   return b;
739   }
740   }
741
742
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 750

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
749   address tokenB,
750   uint amountADesired,
751   uint amountBDesired,
752   uint amountAMin,
753   uint amountBMin,
754
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 990

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
989
990   marketingWallet = address(0x6bd7f652BB574c6f3336b45448c1b4b2d09CF777); // set as
marketing wallet
991   stakingWallet = address(_owner); // set as dev wallet
992   teamWallet = address(0x2ea8EE42dA22f5208905B55F15C66EdA3BFDcfA6); // set as
teamWallet
993
994
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 991

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
990   marketingWallet = address(0x6bd7f652BB574c6f3336b45448c1b4b2d09CF777); // set as
marketing wallet
991   stakingWallet = address(_owner); // set as dev wallet
992   teamWallet = address(0x2ea8EE42dA22f5208905B55F15C66EdA3BFDcfA6); // set as
teamWallet
993
994
995
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 997

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
996
997   //Adding Variables for all the routers for easier deployment for our customers.
998   if (block.chainid == 56) {
999   currentRouter = 0x10ED43C718714eb63d5aA57B78B54704E256024E; // PCS Router
1000   } else if (block.chainid == 97) {
1001
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1000

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
999   currentRouter = 0x10ED43C718714eb63d5aA57B78B54704E256024E; // PCS Router
1000   } else if (block.chainid == 97) {
1001   currentRouter = 0xD99D1c33F9fC3444f8101754aBC46c52416550D1; // PCS Testnet
1002   } else if (block.chainid == 43114) {
1003   currentRouter = 0x60aE616a2155Ee3d9A68541Ba4544862310933d4; //Avax Mainnet
1004
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1004

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1003   currentRouter = 0x60aE616a2155Ee3d9A68541Ba4544862310933d4; //Avax Mainnet
1004   } else if (block.chainid == 137) {
1005   currentRouter = 0xa5E0829CaCEd8fFDD4De3c43696c57F7D7A678ff; //Polygon Ropsten
1006   } else if (block.chainid == 250) {
1007   currentRouter = 0xF491e7B69E4244ad4002BC14e878a34207E38c29; //SpookySwap FTM
1008
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 1088

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1087
1088   function decimals() public view override returns (uint8) {
1089   return _decimals;
1090   }
1091
1092
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1097

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1096
1097   // only use this to disable swapback and send tax in form of tokens
1098   function updateRescueSwap(bool enabled) external onlyOwner(){
1099   rescueSwap = enabled;
1100   }
1101
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1098

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1097   // only use this to disable swapback and send tax in form of tokens
1098   function updateRescueSwap(bool enabled) external onlyOwner(){
1099   rescueSwap = enabled;
1100   }
1101
1102
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1099

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1098   function updateRescueSwap(bool enabled) external onlyOwner(){
1099   rescueSwap = enabled;
1100   }
1101
1102   function setStakingEnabled(bool enabled) external onlyOwner(){
1103
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1103

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1102   function setStakingEnabled(bool enabled) external onlyOwner(){
1103   stakingEnabled = enabled;
1104   }
1105
1106   function updateBuyFees(uint256 _marketingFee, uint256 _liquidityFee, uint256
_devFee, uint256 _teamFee) external onlyOwner {
1107
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1105

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1104   }
1105
1106   function updateBuyFees(uint256 _marketingFee, uint256 _liquidityFee, uint256
_devFee, uint256 _teamFee) external onlyOwner {
1107   buyMarketingFee = _marketingFee;
1108   buyLiquidityFee = _liquidityFee;
1109
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1106

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1105
1106   function updateBuyFees(uint256 _marketingFee, uint256 _liquidityFee, uint256
_devFee, uint256 _teamFee) external onlyOwner {
1107   buyMarketingFee = _marketingFee;
1108   buyLiquidityFee = _liquidityFee;
1109   buyStakingFee = _devFee;
1110
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1127

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1126   transferLiquidityFee = _liquidityFee;
1127   transferStakingFee = _devFee;
1128   transferTeamFee = _teamFee;
1129   transferTotalFees = transferMarketingFee + transferLiquidityFee +
transferStakingFee + transferTeamFee;
1130   require(transferTotalFees <= 11, "Must keep fees at 11% or less");
1131
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1138

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1137
1138   function setAutomatedMarketMakerPair(address pair, bool value) external onlyOwner
{
1139   require(pair != uniswapV2Pair, "The pair cannot be removed from
automatedMarketMakerPairs");
1140
1141   _setAutomatedMarketMakerPair(pair, value);
1142
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1150

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1149
1150   function updateMarketingWallet(address newMarketingWallet) external onlyOwner {
1151   emit marketingWalletUpdated(newMarketingWallet, marketingWallet);
1152   marketingWallet = newMarketingWallet;
1153   }
1154
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1222

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1221   !_isExcludedFromFees[from] &&
1222   !_isExcludedFromFees[to]
1223   ) {
1224   swapping = true;
1225
1226
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1239

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1238   uint256 fees = 0;
1239   // only take fees on buys/sells, do not take on wallet transfers
1240   if(takeFee){
1241
1242   if (automatedMarketMakerPairs[to]){
1243
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1260

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1259   }
1260   } else if (transferTotalFees > 0){
1261   fees = amount.mul(transferTotalFees).div(100);
1262   tokensForLiquidity += fees * transferLiquidityFee / transferTotalFees;
1263   tokensForStaking += fees * transferStakingFee / transferTotalFees;
1264
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1261

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1260   } else if (transferTotalFees > 0){
1261   fees = amount.mul(transferTotalFees).div(100);
1262   tokensForLiquidity += fees * transferLiquidityFee / transferTotalFees;
1263   tokensForStaking += fees * transferStakingFee / transferTotalFees;
1264   tokensForMarketing += fees * transferMarketingFee / transferTotalFees;
1265
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1262

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1261   fees = amount.mul(transferTotalFees).div(100);
1262   tokensForLiquidity += fees * transferLiquidityFee / transferTotalFees;
1263   tokensForStaking += fees * transferStakingFee / transferTotalFees;
1264   tokensForMarketing += fees * transferMarketingFee / transferTotalFees;
1265   tokensForTeam += fees * transferTeamFee / transferTotalFees;
1266
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1263

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1262   tokensForLiquidity += fees * transferLiquidityFee / transferTotalFees;
1263   tokensForStaking += fees * transferStakingFee / transferTotalFees;
1264   tokensForMarketing += fees * transferMarketingFee / transferTotalFees;
1265   tokensForTeam += fees * transferTeamFee / transferTotalFees;
1266   }
1267
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1264

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1263   tokensForStaking += fees * transferStakingFee / transferTotalFees;
1264   tokensForMarketing += fees * transferMarketingFee / transferTotalFees;
1265   tokensForTeam += fees * transferTeamFee / transferTotalFees;
1266   }
1267
1268
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1269

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1268   if(fees > 0){
1269   super._transfer(from, address(this), fees);
1270   }
1271
1272   amount -= fees;
1273
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1270

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1269   super._transfer(from, address(this), fees);
1270   }
1271
1272   amount -= fees;
1273   }
1274
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1275

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1274
1275   super._transfer(from, to, amount);
1276   }
1277
1278   function swapTokensForEth(uint256 tokenAmount) private {
1279
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1278

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1277
1278   function swapTokensForEth(uint256 tokenAmount) private {
1279
1280   // generate the uniswap pair path of token -> weth
1281   address[] memory path = new address[](2);
1282
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1280

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1279
1280   // generate the uniswap pair path of token -> weth
1281   address[] memory path = new address[](2);
1282   path[0] = address(this);
1283   path[1] = uniswapV2Router.WETH();
1284
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1281

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1280   // generate the uniswap pair path of token -> weth
1281   address[] memory path = new address[](2);
1282   path[0] = address(this);
1283   path[1] = uniswapV2Router.WETH();
1284
1285
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1285

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1284
1285   _approve(address(this), address(uniswapV2Router), tokenAmount);
1286
1287   // make the swap
1288   uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
1289
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1287

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1286
1287   // make the swap
1288   uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
1289   tokenAmount,
1290   0, // accept any amount of ETH
1291
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1288

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1287   // make the swap
1288   uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
1289   tokenAmount,
1290   0, // accept any amount of ETH
1291   path,
1292
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1290

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1289   tokenAmount,
1290   0, // accept any amount of ETH
1291   path,
1292   address(this),
1293   block.timestamp
1294
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1291

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1290   0, // accept any amount of ETH
1291   path,
1292   address(this),
1293   block.timestamp
1294   );
1295
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1293

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1292   address(this),
1293   block.timestamp
1294   );
1295
1296   }
1297
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1294

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1293   block.timestamp
1294   );
1295
1296   }
1297
1298
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1294

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1293   block.timestamp
1294   );
1295
1296   }
1297
1298
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 1300

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1299   // approve token transfer to cover all possible scenarios
1300   _approve(address(this), address(uniswapV2Router), tokenAmount);
1301
1302   // add the liquidity
1303   uniswapV2Router.addLiquidityETH{value: ethAmount}(
1304
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1344

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1343
1344   // Halve the amount of liquidity tokens
1345   uint256 liquidityTokens = contractBalance * tokensForLiquidity / totalTokensToSwap
/ 2;
1346   uint256 stakingTokens = contractBalance * tokensForStaking / totalTokensToSwap;
1347   uint256 amountToSwapForETH =
contractBalance.sub(liquidityTokens).sub(stakingTokens);
1348
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1345

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1344   // Halve the amount of liquidity tokens
1345   uint256 liquidityTokens = contractBalance * tokensForLiquidity / totalTokensToSwap
/ 2;
1346   uint256 stakingTokens = contractBalance * tokensForStaking / totalTokensToSwap;
1347   uint256 amountToSwapForETH =
contractBalance.sub(liquidityTokens).sub(stakingTokens);
1348
1349
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1346

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1345   uint256 liquidityTokens = contractBalance * tokensForLiquidity / totalTokensToSwap
/ 2;
1346   uint256 stakingTokens = contractBalance * tokensForStaking / totalTokensToSwap;
1347   uint256 amountToSwapForETH =
contractBalance.sub(liquidityTokens).sub(stakingTokens);
1348
1349   uint256 initialETHBalance = address(this).balance;
1350
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1347

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1346   uint256 stakingTokens = contractBalance * tokensForStaking / totalTokensToSwap;
1347   uint256 amountToSwapForETH =
contractBalance.sub(liquidityTokens).sub(stakingTokens);
1348
1349   uint256 initialETHBalance = address(this).balance;
1350
1351
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1356

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1355
1356   uint256 ethForMarketing =
ethBalance.mul(tokensForMarketing).div(totalTokensToSwap);
1357   uint256 ethForTeam = ethBalance.mul(tokensForTeam).div(totalTokensToSwap);
1358
1359   uint256 ethForLiquidity = ethBalance - ethForMarketing - ethForTeam;
1360
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1356

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1355
1356   uint256 ethForMarketing =
ethBalance.mul(tokensForMarketing).div(totalTokensToSwap);
1357   uint256 ethForTeam = ethBalance.mul(tokensForTeam).div(totalTokensToSwap);
1358
1359   uint256 ethForLiquidity = ethBalance - ethForMarketing - ethForTeam;
1360
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1356

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1355
1356   uint256 ethForMarketing =
ethBalance.mul(tokensForMarketing).div(totalTokensToSwap);
1357   uint256 ethForTeam = ethBalance.mul(tokensForTeam).div(totalTokensToSwap);
1358
1359   uint256 ethForLiquidity = ethBalance - ethForMarketing - ethForTeam;
1360
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1363

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1362   tokensForMarketing = 0;
1363   tokensForStaking = 0;
1364   tokensForTeam = 0;
1365
1366   (success,) = address(teamWallet).call{value: ethForTeam}("");
1367
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1366

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1365
1366   (success,) = address(teamWallet).call{value: ethForTeam}("");
1367
1368   if(liquidityTokens > 0 && ethForLiquidity > 0){
1369   addLiquidity(liquidityTokens, ethForLiquidity);
1370
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1383

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1382    }
1383    }
1384
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 47

low SEVERITY

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.

Source File

- LunarRabbit.sol

Locations

```
46   function totalSupply() external view returns (uint);
47   function balanceOf(address owner) external view returns (uint);
48   function allowance(address owner, address spender) external view returns (uint);
49
50   function approve(address spender, uint value) external returns (bool);
51
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1088

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1087
1088   function decimals() public view override returns (uint8) {
1089   return _decimals;
1090   }
1091
1092
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1092

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1091
1092   // only use to disable contract sales if absolutely necessary (emergency use only)
1093   function updateSwapEnabled(bool enabled) external onlyOwner(){
1094   swapEnabled = enabled;
1095   }
1096
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1307

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1306   0, // slippage is unavoidable
1307   0, // slippage is unavoidable
1308   deadAddress,
1309   block.timestamp
1310   );
1311
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1308

low SEVERITY

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

Source File

- LunarRabbit.sol

Locations

```
1307   0, // slippage is unavoidable
1308   deadAddress,
1309   block.timestamp
1310   );
1311   }
1312
```

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.