Kiba Inu

Smart Contract Audit Report

SYSFIXED

05 Mar 2022

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
Kiba Inu	KIBA	Binance Smart Chain

Addresses

Contract address	0xc3afde95b6eb9ba8553cdaea6645d45fb3a7faf5
Contract deployer address	0x2c5B9dd42d0510C43f1d6d672bD56A7DE0716c91

Project Website

https://kibainu.com/#/swap

Codebase

https://bscscan.com/address/0xc3afde95b6eb9ba8553cdaea6645d45fb3a7faf5#contracts

SUMMARY

Kiba Inu is a meme coin that has evolved into a meme and utility coin with the introduction of KibaSwap.

Contract Summary

Documentation Quality

Kiba Inu provides a very good documentation with standard of solidity base code.

- The technical description is provided clearly and structured and also dont have any high risk issue.

Code Quality

The Overall quality of the basecode is standard.

- Standard solidity basecode and rules are already followed by Kiba Inu with the discovery of several low issues.

Test Coverage

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

Audit Findings Summary

- SWC-100 SWC-108 | Explicitly define visibility for all state variables on lines 916.
- SWC-101 | It is recommended to use vetted safe math libraries for arithmetic operations consistently on lines 469, 501, 524, 525, 560, 596, 662, 666, 678, 685, 694, 970, 970, 972, 972, 974, 974, 979, 979, 984, 984, 1051, 1051, 1052, 1052, 1058, 1058, 1058, 1059, 1059, 1063, 1063, 1063, 1064, 1064, 1080, 1080, 1090, 1090, 1168, 1176, 1182, 1194, 1197, 1197, 1224, 1242, 1242, 1242, 1243, 1243, 1243, 1244, 1244, 1244, 1249, 1249, 1249, 1250, 1250, 1250, 1251, 1251, 1251, 1258, 1303, 1303, 1308, 1309, 1313, 1313, 1313, 1326, 1326 and 1375.
- 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 1268 and 1269.
- SWC-115 | tx.origin should not be used for authorization, use msg.sender instead on lines 1160 and 1161.
- SWC-120 | It is recommended to use external sources of randomness via oracles on lines 1021, 1160, 1161 and 1182.

CONCLUSION

We have audited the Kiba Inu project released on March 2022 to discover issues and identify potential security vulnerabilities in Kiba Inu Project. This process is used to find technical issues and security loopholes which might be found in the smart contract.

The security audit report provides satisfactory results with low-risk issues.

The Kiba Inu smart contract code issues do not pose a considerable risk. The writing of the contract is close to the standard of writing contracts in general. The low-risk issues found are some use of "tx.origin" as a part of authorization control, state variable visibility is not set, out of bounds array access, the potential use of "block.number" as a source of randomness, and out-of-bounds array access which the index access expression can cause an exception in case of the use of an invalid array index value. It is best practice to set the visibility of state variables explicitly. The default visibility for "launchedAt" is internal. Other possible visibility settings are public and private. 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. Be aware that using these variables introduces a certain level of trust into miners, and tx.origin should not be used for authorization. Use msg.sender instead.

AUDIT RESULT

Article	Category	Description	Result
Default Visibility	SWC-100 SWC-108	Functions and state variables visibility should be set explicitly. Visibility levels should be specified consciously.	ISSUE FOUND
Integer Overflow and Underflow	SWC-101	If unchecked math is used, all math operations should be safe from overflows and underflows.	ISSUE FOUND
Outdated Compiler Version	SWC-102	It is recommended to use a recent version of the Solidity compiler.	PASS
Floating Pragma	SWC-103	Contracts should be deployed with the same compiler version and flags that they have been tested thoroughly.	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.	ISSUE FOUND
Block values as a proxy for time	SWC-116	Block numbers should not be used for time calculations.	PASS
Signature Unique ID	SWC-117 SWC-121 SWC-122	Signed messages should always have a unique id. A transaction hash should not be used as a unique id.	PASS
Incorrect Constructor Name	SWC-118	Constructors are special functions that are called only once during the contract creation.	PASS
Shadowing State Variable	SWC-119	State variables should not be shadowed.	PASS
Weak Sources of Randomness	SWC-120	Random values should never be generated from Chain Attributes or be predictable.	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	Friday Mar 04 2022 18:54:54 GMT+0000 (Coordinated Universal Time)
Finished	Saturday Mar 05 2022 19:20:03 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	Kibalnu.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
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-108	STATE VARIABLE VISIBILITY IS NOT SET.	low	acknowledged
SWC-115	USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.	low	acknowledged
SWC-115	USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-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 469

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
468   function add(uint256 a, uint256 b) internal pure returns (uint256) {
469   uint256 c = a + b;
470   require(c >= a, "SafeMath: addition overflow");
471
472   return c;
473
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 501

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
500   require(b <= a, errorMessage);
501   uint256 c = a - b;
502
503   return c;
504   }
505
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 524

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
523
524   uint256 c = a * b;
525   require(c / a == b, "SafeMath: multiplication overflow");
526
527   return c;
528
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 525

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
524   uint256 c = a * b;
525   require(c / a == b, "SafeMath: multiplication overflow");
526
527   return c;
528   }
529
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 560

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
559   require(b > 0, errorMessage);
560   uint256 c = a / b;
561   // assert(a == b * c + a % b); // There is no case in which this doesn't hold
562
563   return c;
564
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 596

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
595   require(b != 0, errorMessage);
596   return a % b;
597   }
598   }
599
600
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 662

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
661   function mul(int256 a, int256 b) internal pure returns (int256) {
662   int256 c = a * b;
663
664   // Detect overflow when multiplying MIN_INT256 with -1
665   require(c != MIN_INT256 || (a & MIN_INT256) != (b & MIN_INT256));
666
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 666

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
665   require(c != MIN_INT256 || (a & MIN_INT256) != (b & MIN_INT256));
666   require((b == 0) || (c / b == a));
667   return c;
668   }
669
670
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 678

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
677   // Solidity already throws when dividing by 0.
678   return a / b;
679   }
680
681   /**
682
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 685

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
684   function sub(int256 a, int256 b) internal pure returns (int256) {
685   int256 c = a - b;
686   require((b >= 0 && c <= a) || (b < 0 && c > a));
687   return c;
688   }
689
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 694

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
693   function add(int256 a, int256 b) internal pure returns (int256) {
694   int256 c = a + b;
695   require((b >= 0 && c >= a) || (b < 0 && c < a));
696   return c;
697   }
698
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 970

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
969
970   uint256 totalSupply = 1 * 1e12 * 1e18;
971
972   maxTransactionAmount = totalSupply * 2 / 1000; // 0.2% maxTransactionAmountTxn
973   maxWallet = totalSupply; // No Max Wallet On Launch
974
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 970

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
969
970   uint256 totalSupply = 1 * 1e12 * 1e18;
971
972   maxTransactionAmount = totalSupply * 2 / 1000; // 0.2% maxTransactionAmountTxn
973   maxWallet = totalSupply; // No Max Wallet On Launch
974
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 972

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
971
972   maxTransactionAmount = totalSupply * 2 / 1000; // 0.2% maxTransactionAmountTxn
973   maxWallet = totalSupply; // No Max Wallet On Launch
974   swapTokensAtAmount = totalSupply * 5 / 10000; // 0.05% swap wallet
975
976
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 972

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
971
972   maxTransactionAmount = totalSupply * 2 / 1000; // 0.2% maxTransactionAmountTxn
973   maxWallet = totalSupply; // No Max Wallet On Launch
974   swapTokensAtAmount = totalSupply * 5 / 10000; // 0.05% swap wallet
975
976
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 974

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
973   maxWallet = totalSupply; // No Max Wallet On Launch
974   swapTokensAtAmount = totalSupply * 5 / 10000; // 0.05% swap wallet
975
976   buyMarketingFee = _buyMarketingFee;
977   buyLiquidityFee = _buyLiquidityFee;
978
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 974

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
973   maxWallet = totalSupply; // No Max Wallet On Launch
974   swapTokensAtAmount = totalSupply * 5 / 10000; // 0.05% swap wallet
975
976   buyMarketingFee = _buyMarketingFee;
977   buyLiquidityFee = _buyLiquidityFee;
978
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 979

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
978   buyDevFee = _buyDevFee;
979   buyTotalFees = buyMarketingFee + buyLiquidityFee + buyDevFee;
980
981   sellMarketingFee = _sellMarketingFee;
982   sellLiquidityFee = _sellLiquidityFee;
983
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 979

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
978   buyDevFee = _buyDevFee;
979   buyTotalFees = buyMarketingFee + buyLiquidityFee + buyDevFee;
980
981   sellMarketingFee = _sellMarketingFee;
982   sellLiquidityFee = _sellLiquidityFee;
983
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 984

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
983   sellDevFee = _sellDevFee;
984   sellTotalFees = sellMarketingFee + sellLiquidityFee + sellDevFee;
985
986   earlySellLiquidityFee = _earlySellLiquidityFee;
987   earlySellMarketingFee = _earlySellMarketingFee;
988
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 984

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
983   sellDevFee = _sellDevFee;
984   sellTotalFees = sellMarketingFee + sellLiquidityFee + sellDevFee;
985
986   earlySellLiquidityFee = _earlySellLiquidityFee;
987   earlySellMarketingFee = _earlySellMarketingFee;
988
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1051

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1050   function updateSwapTokensAtAmount(uint256 newAmount) external onlyOwner returns
(bool){
1051   require(newAmount >= totalSupply() * 1 / 100000, "Swap amount cannot be lower than
0.001% total supply.");
1052   require(newAmount <= totalSupply() * 5 / 1000, "Swap amount cannot be higher than
0.5% total supply.");
1053   swapTokensAtAmount = newAmount;
1054   return true;
1055
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1051

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1050   function updateSwapTokensAtAmount(uint256 newAmount) external onlyOwner returns
(bool){
1051   require(newAmount >= totalSupply() * 1 / 100000, "Swap amount cannot be lower than
0.001% total supply.");
1052   require(newAmount <= totalSupply() * 5 / 1000, "Swap amount cannot be higher than
0.5% total supply.");
1053   swapTokensAtAmount = newAmount;
1054   return true;
1055
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1052

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1051   require(newAmount >= totalSupply() * 1 / 100000, "Swap amount cannot be lower than
0.001% total supply.");
1052   require(newAmount <= totalSupply() * 5 / 1000, "Swap amount cannot be higher than
0.5% total supply.");
1053   swapTokensAtAmount = newAmount;
1054   return true;
1055   }
1056
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1052

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1051   require(newAmount >= totalSupply() * 1 / 100000, "Swap amount cannot be lower than
0.001% total supply.");
1052   require(newAmount <= totalSupply() * 5 / 1000, "Swap amount cannot be higher than
0.5% total supply.");
1053   swapTokensAtAmount = newAmount;
1054   return true;
1055   }
1056
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1058

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1057   function updateMaxTxnAmount(uint256 newNum) external onlyOwner {
1058   require(newNum >= (totalSupply() * 1 / 1000)/1e18, "Cannot set
maxTransactionAmount lower than 0.1%");
1059   maxTransactionAmount = newNum * (10**18);
1060   }
1061
1062
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1058

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1057   function updateMaxTxnAmount(uint256 newNum) external onlyOwner {
1058   require(newNum >= (totalSupply() * 1 / 1000)/1e18, "Cannot set
maxTransactionAmount lower than 0.1%");
1059   maxTransactionAmount = newNum * (10**18);
1060   }
1061
1062
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1058

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1057   function updateMaxTxnAmount(uint256 newNum) external onlyOwner {
1058   require(newNum >= (totalSupply() * 1 / 1000)/1e18, "Cannot set
maxTransactionAmount lower than 0.1%");
1059   maxTransactionAmount = newNum * (10**18);
1060   }
1061
1062
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1059

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1058   require(newNum >= (totalSupply() * 1 / 1000)/1e18, "Cannot set
maxTransactionAmount lower than 0.1%");
1059   maxTransactionAmount = newNum * (10**18);
1060   }
1061
1062   function updateMaxWalletAmount(uint256 newNum) external onlyOwner {
1063
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1059

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1058   require(newNum >= (totalSupply() * 1 / 1000)/1e18, "Cannot set
maxTransactionAmount lower than 0.1%");
1059   maxTransactionAmount = newNum * (10**18);
1060   }
1061
1062   function updateMaxWalletAmount(uint256 newNum) external onlyOwner {
1063
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1063

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1062   function updateMaxWalletAmount(uint256 newNum) external onlyOwner {
1063   require(newNum >= (totalSupply() * 5 / 1000)/1e18, "Cannot set maxWallet lower
than 0.5%");
1064   maxWallet = newNum * (10**18);
1065   }
1066
1067
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1063

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1062   function updateMaxWalletAmount(uint256 newNum) external onlyOwner {
1063   require(newNum >= (totalSupply() * 5 / 1000)/1e18, "Cannot set maxWallet lower
than 0.5%");
1064   maxWallet = newNum * (10**18);
1065   }
1066
1067
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1063

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1062   function updateMaxWalletAmount(uint256 newNum) external onlyOwner {
1063   require(newNum >= (totalSupply() * 5 / 1000)/1e18, "Cannot set maxWallet lower
than 0.5%");
1064   maxWallet = newNum * (10**18);
1065   }
1066
1067
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1064

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1063   require(newNum >= (totalSupply() * 5 / 1000)/1e18, "Cannot set maxWallet lower
than 0.5%");
1064   maxWallet = newNum * (10**18);
1065   }
1066
1067   function excludeFromMaxTransaction(address updAds, bool isEx) public onlyOwner {
1068
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1064

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1063   require(newNum >= (totalSupply() * 5 / 1000)/1e18, "Cannot set maxWallet lower
than 0.5%");
1064   maxWallet = newNum * (10**18);
1065   }
1066
1067   function excludeFromMaxTransaction(address updAds, bool isEx) public onlyOwner {
1068
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1080

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1079   buyDevFee = _devFee;
1080   buyTotalFees = buyMarketingFee + buyLiquidityFee + buyDevFee;
1081   require(buyTotalFees <= 20, "Must keep fees at 20% or less");
1082   }
1083
1084
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1080

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1079   buyDevFee = _devFee;
1080   buyTotalFees = buyMarketingFee + buyLiquidityFee + buyDevFee;
1081   require(buyTotalFees <= 20, "Must keep fees at 20% or less");
1082   }
1083
1084
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1090

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1089   earlySellMarketingFee = _earlySellMarketingFee;
1090   sellTotalFees = sellMarketingFee + sellLiquidityFee + sellDevFee;
1091   require(sellTotalFees <= 25, "Must keep fees at 25% or less");
1092   }
1093
1094
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1090

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1089   earlySellMarketingFee = _earlySellMarketingFee;
1090   sellTotalFees = sellMarketingFee + sellLiquidityFee + sellDevFee;
1091   require(sellTotalFees <= 25, "Must keep fees at 25% or less");
1092   }
1093
1094
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1168

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1167   require(amount <= maxTransactionAmount, "Buy transfer amount exceeds the
maxTransactionAmount.");
1168   require(amount + balanceOf(to) <= maxWallet, "Max wallet exceeded");
1169   }
1170
1171   //when sell
1172
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1176

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1175   else if(!_isExcludedMaxTransactionAmount[to]){
1176   require(amount + balanceOf(to) <= maxWallet, "Max wallet exceeded");
1177   }
1178   }
1179   }
1180
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1182

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1181   // anti bot logic
1182   if (block.number <= (launchedAt + 1) &&
1183   to != uniswapV2Pair &&
1184   to != address(0x10ED43C718714eb63d5aA57B78B54704E256024E)
1185   ) {
1186
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1194

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1193   if (_holderFirstBuyTimestamp[from] != 0 &&
1194   (_holderFirstBuyTimestamp[from] + (24 hours) >= block.timestamp))  {
1195   sellLiquidityFee = earlySellLiquidityFee;
1196   sellMarketingFee = earlySellMarketingFee;
1197   sellTotalFees = sellMarketingFee + sellLiquidityFee + sellDevFee;
1198
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1197

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1196   sellMarketingFee = earlySellMarketingFee;
1197   sellTotalFees = sellMarketingFee + sellLiquidityFee + sellDevFee;
1198   }
1199   } else {
1200   if (_holderFirstBuyTimestamp[to] == 0) {
1201
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1197

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1196   sellMarketingFee = earlySellMarketingFee;
1197   sellTotalFees = sellMarketingFee + sellLiquidityFee + sellDevFee;
1198   }
1199   } else {
1200   if (_holderFirstBuyTimestamp[to] == 0) {
1201
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1224

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1223
1224   if(!swapping && automatedMarketMakerPairs[to] && lpBurnEnabled && block.timestamp
>= lastLpBurnTime + lpBurnFrequency && !_isExcludedFromFees[from]){
1225   autoBurnLiquidityPairTokens();
1226   }
1227
1228
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1242

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1241   fees = amount.mul(sellTotalFees).div(100);
1242   tokensForLiquidity += fees * sellLiquidityFee / sellTotalFees;
1243   tokensForDev += fees * sellDevFee / sellTotalFees;
1244   tokensForMarketing += fees * sellMarketingFee / sellTotalFees;
1245   }
1246
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1242

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1241   fees = amount.mul(sellTotalFees).div(100);
1242   tokensForLiquidity += fees * sellLiquidityFee / sellTotalFees;
1243   tokensForDev += fees * sellDevFee / sellTotalFees;
1244   tokensForMarketing += fees * sellMarketingFee / sellTotalFees;
1245   }
1246
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1242

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1241   fees = amount.mul(sellTotalFees).div(100);
1242   tokensForLiquidity += fees * sellLiquidityFee / sellTotalFees;
1243   tokensForDev += fees * sellDevFee / sellTotalFees;
1244   tokensForMarketing += fees * sellMarketingFee / sellTotalFees;
1245   }
1246
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1243

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1242   tokensForLiquidity += fees * sellLiquidityFee / sellTotalFees;
1243   tokensForDev += fees * sellDevFee / sellTotalFees;
1244   tokensForMarketing += fees * sellMarketingFee / sellTotalFees;
1245   }
1246   // on buy
1247
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1243

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1242   tokensForLiquidity += fees * sellLiquidityFee / sellTotalFees;
1243   tokensForDev += fees * sellDevFee / sellTotalFees;
1244   tokensForMarketing += fees * sellMarketingFee / sellTotalFees;
1245   }
1246   // on buy
1247
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1243

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1242   tokensForLiquidity += fees * sellLiquidityFee / sellTotalFees;
1243   tokensForDev += fees * sellDevFee / sellTotalFees;
1244   tokensForMarketing += fees * sellMarketingFee / sellTotalFees;
1245   }
1246   // on buy
1247
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1244

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1243   tokensForDev += fees * sellDevFee / sellTotalFees;
1244   tokensForMarketing += fees * sellMarketingFee / sellTotalFees;
1245   }
1246   // on buy
1247   else if(automatedMarketMakerPairs[from] && buyTotalFees > 0) {
1248
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1244

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1243   tokensForDev += fees * sellDevFee / sellTotalFees;
1244   tokensForMarketing += fees * sellMarketingFee / sellTotalFees;
1245   }
1246   // on buy
1247   else if(automatedMarketMakerPairs[from] && buyTotalFees > 0) {
1248
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1244

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1243   tokensForDev += fees * sellDevFee / sellTotalFees;
1244   tokensForMarketing += fees * sellMarketingFee / sellTotalFees;
1245   }
1246   // on buy
1247   else if(automatedMarketMakerPairs[from] && buyTotalFees > 0) {
1248
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1249

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1248   fees = amount.mul(buyTotalFees).div(100);
1249   tokensForLiquidity += fees * buyLiquidityFee / buyTotalFees;
1250   tokensForDev += fees * buyDevFee / buyTotalFees;
1251   tokensForMarketing += fees * buyMarketingFee / buyTotalFees;
1252   }
1253
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1249

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1248   fees = amount.mul(buyTotalFees).div(100);
1249   tokensForLiquidity += fees * buyLiquidityFee / buyTotalFees;
1250   tokensForDev += fees * buyDevFee / buyTotalFees;
1251   tokensForMarketing += fees * buyMarketingFee / buyTotalFees;
1252   }
1253
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1249

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1248   fees = amount.mul(buyTotalFees).div(100);
1249   tokensForLiquidity += fees * buyLiquidityFee / buyTotalFees;
1250   tokensForDev += fees * buyDevFee / buyTotalFees;
1251   tokensForMarketing += fees * buyMarketingFee / buyTotalFees;
1252   }
1253
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1250

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1249   tokensForLiquidity += fees * buyLiquidityFee / buyTotalFees;
1250   tokensForDev += fees * buyDevFee / buyTotalFees;
1251   tokensForMarketing += fees * buyMarketingFee / buyTotalFees;
1252   }
1253
1254
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1250

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1249   tokensForLiquidity += fees * buyLiquidityFee / buyTotalFees;
1250   tokensForDev += fees * buyDevFee / buyTotalFees;
1251   tokensForMarketing += fees * buyMarketingFee / buyTotalFees;
1252   }
1253
1254
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1250

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1249   tokensForLiquidity += fees * buyLiquidityFee / buyTotalFees;
1250   tokensForDev += fees * buyDevFee / buyTotalFees;
1251   tokensForMarketing += fees * buyMarketingFee / buyTotalFees;
1252   }
1253
1254
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1251

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1250   tokensForDev += fees * buyDevFee / buyTotalFees;
1251   tokensForMarketing += fees * buyMarketingFee / buyTotalFees;
1252   }
1253
1254   if(fees > 0){
1255
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1251

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1250   tokensForDev += fees * buyDevFee / buyTotalFees;
1251   tokensForMarketing += fees * buyMarketingFee / buyTotalFees;
1252   }
1253
1254   if(fees > 0){
1255
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1251

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1250   tokensForDev += fees * buyDevFee / buyTotalFees;
1251   tokensForMarketing += fees * buyMarketingFee / buyTotalFees;
1252   }
1253
1254   if(fees > 0){
1255
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 1258

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1257
1258   amount -= fees;
1259   }
1260
1261   super._transfer(from, to, amount);
1262
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1303

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1302   uint256 contractBalance = balanceOf(address(this));
1303   uint256 totalTokensToSwap = tokensForLiquidity + tokensForMarketing +
tokensForDev;
1304   bool success;
1305
1306   if(contractBalance == 0 || totalTokensToSwap == 0) {return;}
1307
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1303

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1302   uint256 contractBalance = balanceOf(address(this));
1303   uint256 totalTokensToSwap = tokensForLiquidity + tokensForMarketing +
tokensForDev;
1304   bool success;
1305
1306   if(contractBalance == 0 || totalTokensToSwap == 0) {return;}
1307
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1308

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1307
1308   if(contractBalance > swapTokensAtAmount * 20){
1309   contractBalance = swapTokensAtAmount * 20;
1310   }
1311
1312
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1309

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1308   if(contractBalance > swapTokensAtAmount * 20){
1309   contractBalance = swapTokensAtAmount * 20;
1310   }
1311
1312   // Halve the amount of liquidity tokens
1313
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1313

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1312   // Halve the amount of liquidity tokens
1313   uint256 liquidityTokens = contractBalance * tokensForLiquidity / totalTokensToSwap
/ 2;
1314   uint256 amountToSwapForETH = contractBalance.sub(liquidityTokens);
1315
1316   uint256 initialETHBalance = address(this).balance;
1317
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1313

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1312   // Halve the amount of liquidity tokens
1313   uint256 liquidityTokens = contractBalance * tokensForLiquidity / totalTokensToSwap
/ 2;
1314   uint256 amountToSwapForETH = contractBalance.sub(liquidityTokens);
1315
1316   uint256 initialETHBalance = address(this).balance;
1317
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1313

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1312   // Halve the amount of liquidity tokens
1313   uint256 liquidityTokens = contractBalance * tokensForLiquidity / totalTokensToSwap
/ 2;
1314   uint256 amountToSwapForETH = contractBalance.sub(liquidityTokens);
1315
1316   uint256 initialETHBalance = address(this).balance;
1317
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1326

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1325
1326   uint256 ethForLiquidity = ethBalance - ethForMarketing - ethForDev;
1327
1328
1329   tokensForLiquidity = 0;
1330
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1326

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1325
1326   uint256 ethForLiquidity = ethBalance - ethForMarketing - ethForDev;
1327
1328
1329   tokensForLiquidity = 0;
1330
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1375

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1374   function manualBurnLiquidityPairTokens(uint256 percent) external onlyOwner returns
(bool){
1375   require(block.timestamp > lastManualLpBurnTime + manualBurnFrequency , "Must wait
for cooldown to finish");
1376   require(percent <= 1000, "May not nuke more than 10% of tokens in LP");
1377   lastManualLpBurnTime = block.timestamp;
1378
1379
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 916

low SEVERITY

It is best practice to set the visibility of state variables explicitly. The default visibility for "launchedAt" is internal. Other possible visibility settings are public and private.

Source File

- KibaInu.sol

Locations

```
915   // block number of opened trading
916   uint256 launchedAt;
917
918   /******************/
919
920
```

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

LINE 1160

low SEVERITY

The tx.origin environment variable has been found to influence a control flow decision. Note that using "tx.origin" as a security control might cause a situation where a user inadvertently authorizes a smart contract to perform an action on their behalf. It is recommended to use "msg.sender" instead.

Source File

- KibaInu.sol

Locations

```
1159   if (to != owner() && to != address(uniswapV2Router) && to !=
address(uniswapV2Pair)){
1160   require(_holderLastTransferTimestamp[tx.origin] < block.number, "_transfer::
Transfer Delay enabled.  Only one purchase per block allowed.");
1161   _holderLastTransferTimestamp[tx.origin] = block.number;
1162   }
1163   }
1164
```

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

LINE 1161

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1160   require(_holderLastTransferTimestamp[tx.origin] < block.number, "_transfer::
Transfer Delay enabled.  Only one purchase per block allowed.");
1161   _holderLastTransferTimestamp[tx.origin] = block.number;
1162   }
1163   }
1164
1165
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1268

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1267   address[] memory path = new address[](2);
1268   path[0] = address(this);
1269   path[1] = uniswapV2Router.WETH();
1270
1271   _approve(address(this), address(uniswapV2Router), tokenAmount);
1272
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1269

low SEVERITY

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

Source File

- KibaInu.sol

Locations

```
1268   path[0] = address(this);
1269   path[1] = uniswapV2Router.WETH();
1270
1271   _approve(address(this), address(uniswapV2Router), tokenAmount);
1272
1273
```

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

LINE 1021

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

- KibaInu.sol

Locations

```
1020   lastLpBurnTime = block.timestamp;
1021   launchedAt = block.number;
1022   }
1023
1024   // remove limits after token is stable
1025
```

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

LINE 1160

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

- KibaInu.sol

Locations

```
1159   if (to != owner() && to != address(uniswapV2Router) && to !=
address(uniswapV2Pair)){
1160   require(_holderLastTransferTimestamp[tx.origin] < block.number, "_transfer::
Transfer Delay enabled.  Only one purchase per block allowed.");
1161   _holderLastTransferTimestamp[tx.origin] = block.number;
1162   }
1163   }
1164
```

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

LINE 1161

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

- KibaInu.sol

Locations

```
1160   require(_holderLastTransferTimestamp[tx.origin] < block.number, "_transfer::
Transfer Delay enabled.  Only one purchase per block allowed.");
1161   _holderLastTransferTimestamp[tx.origin] = block.number;
1162   }
1163   }
1164
1165
```

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

LINE 1182

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

- KibaInu.sol

Locations

```
1181   // anti bot logic
1182   if (block.number <= (launchedAt + 1) &&
1183   to != uniswapV2Pair &&
1184   to != address(0x10ED43C718714eb63d5aA57B78B54704E256024E)
1185   ) {
1186
```

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.