Sheikh Inu

Smart Contract Audit Report

SYSFIXED

01 Feb 2023

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
Sheikh Inu	SHINU	Binance Smart Chain

Addresses

Contract address	0xE5b5d4Bea7468B4994FA676949308a79497aa24c
Contract deployer address	0xf5B87F2D9eb0923a8f274c277CC96429D375321f

Project Website

https://sheikhinu.io/

Codebase

https://bscscan.com/address/0xE5b5d4Bea7468B4994FA676949308a79497aa24c#code

SUMMARY

Sheikh Inu $SHINU Marhaba traveller! Welcome to Baba Sheikh's oasis. A heavenly refueling place for every adventurer seeking treasure. Fancy a nice dastarkhan on the brink of the oasis lake? Join us on the trip to the desert treasures: -Experienced Team -Trusted Deployer -Top Backers -BSC #BNB Cult -2023 Bullrun Kickstart.

Contract Summary

Documentation Quality

Sheikh 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 Sheikh Inu 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 198, 220, 245, 274, 275, 404, 404, 405, 405, 406, 406, 407, 407, 440, 440, 472, 482, 493, 516, 523, 527, 540, 549, 555, 564, 564, 571, 575, 575, 595, 596, 596, 597, 603, 604, 604, 605, 612, 612, 661, 661, 669, 677, 704, 714, 723, 723, 724, 724, 725 and 725.
- SWC-103 | Pragma statements can be allowed to float when a contract is intended on lines 11.
- 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 623, 624 and 715.
- SWC-120 | It is recommended to use external sources of randomness via oracles on lines 540 and 689.

CONCLUSION

We have audited the Sheikh Inu project released on January 2023 to discover issues and identify potential security vulnerabilities in Sheikh 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 a satisfactory result with some low-risk issues.

The issues found in the Sheikh Inu smart contract code do not pose a considerable risk. The writing of the contract is close to the standard of writing contracts in general. The low-risk issues found are some arithmetic operation issues, a floating pragma set, weak sources 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. We recommend to Don't using any of those environment variables as sources of randomness and being aware that the use of these variables introduces a certain level of trust in miners.

AUDIT RESULT

Article	Category	Description	Result
Default Visibility	SWC-100 SWC-108	Functions and state variables visibility should be set explicitly. Visibility levels should be specified consciously.	PASS
Integer Overflow and Underflow	SWC-101	If unchecked math is used, all math operations should be safe from overflows and underflows.	ISSUE FOUND
Outdated Compiler Version	SWC-102	It is recommended to use a recent version of the Solidity compiler.	PASS
Floating Pragma	SWC-103	Contracts should be deployed with the same compiler version and flags that they have been tested thoroughly.	ISSUE FOUND
Unchecked Call Return Value	SWC-104	The return value of a message call should be checked.	PASS
Unprotected Ether Withdrawal	SWC-105	Due to missing or insufficient access controls, malicious parties can withdraw from the contract.	PASS
SELFDESTRUCT Instruction	SWC-106	The contract should not be self-destructible while it has funds belonging to users.	PASS
Reentrancy	SWC-107	Check effect interaction pattern should be followed if the code performs recursive call.	PASS
Uninitialized Storage Pointer	SWC-109	Uninitialized local storage variables can point to unexpected storage locations in the contract.	PASS
Assert Violation	SWC-110 SWC-123	Properly functioning code should never reach a failing assert statement.	ISSUE FOUND
Deprecated Solidity Functions	SWC-111	Deprecated built-in functions should never be used.	PASS
Delegate call to Untrusted Callee	SWC-112	Delegatecalls should only be allowed to trusted addresses.	PASS

DoS (Denial of Service)	SWC-113 SWC-128	Execution of the code should never be blocked by a specific contract state unless required.	PASS
Race Conditions	SWC-114	Race Conditions and Transactions Order Dependency should not be possible.	PASS
Authorization through tx.origin	SWC-115	tx.origin should not be used for authorization.	PASS
Block values as a proxy for time	SWC-116	Block numbers should not be used for time calculations.	PASS
Signature Unique ID	SWC-117 SWC-121 SWC-122	Signed messages should always have a unique id. A transaction hash should not be used as a unique id.	PASS
Incorrect Constructor Name	SWC-118	Constructors are special functions that are called only once during the contract creation.	PASS
Shadowing State Variable	SWC-119	State variables should not be shadowed.	PASS
Weak Sources of Randomness	SWC-120	Random values should never be generated from Chain Attributes or be predictable.	ISSUE FOUND
Write to Arbitrary Storage Location	SWC-124	The contract is responsible for ensuring that only authorized user or contract accounts may write to sensitive storage locations.	PASS
Incorrect Inheritance Order	SWC-125	When inheriting multiple contracts, especially if they have identical functions, a developer should carefully specify inheritance in the correct order. The rule of thumb is to inherit contracts from more /general/ to more /specific/.	PASS
Insufficient Gas Griefing	SWC-126	Insufficient gas griefing attacks can be performed on contracts which accept data and use it in a sub-call on another contract.	PASS
Arbitrary Jump Function	SWC-127	As Solidity doesnt support pointer arithmetics, it is impossible to change such variable to an arbitrary value.	PASS

Typographical Error	SWC-129	A typographical error can occur for example when the intent of a defined operation is to sum a number to a variable.	PASS
Override control character	SWC-130	Malicious actors can use the Right-To-Left-Override unicode character to force RTL text rendering and confuse users as to the real intent of a contract.	PASS
Unused variables	SWC-131 SWC-135	Unused variables are allowed in Solidity and they do not pose a direct security issue.	PASS
Unexpected Ether balance	SWC-132	Contracts can behave erroneously when they strictly assume a specific Ether balance.	PASS
Hash Collisions Variable	SWC-133	Using abi.encodePacked() with multiple variable length arguments can, in certain situations, lead to a hash collision.	PASS
Hardcoded gas amount	SWC-134	The transfer() and send() functions forward a fixed amount of 2300 gas.	PASS
Unencrypted Private Data	SWC-136	It is a common misconception that private type variables cannot be read.	PASS

SMART CONTRACT ANALYSIS

Started	Tuesday Jan 31 2023 18:50:56 GMT+0000 (Coordinated Universal Time)
Finished	Wednesday Feb 01 2023 19:26:16 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	SheikhInu.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-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-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 198

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
197   require(currentAllowance >= amount, "BEP20: transfer amount exceeds allowance");
198   _approve(sender, _msgSender(), currentAllowance - amount);
199
200   return true;
201   }
202
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 220

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
219   {
220   _approve(_msgSender(), spender, _allowances[_msgSender()][spender] + addedValue);
221   return true;
222   }
223
224
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 245

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
244   require(currentAllowance >= subtractedValue, "BEP20: decreased allowance below
zero");
245   _approve(_msgSender(), spender, currentAllowance - subtractedValue);
246
247   return true;
248   }
249
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 274

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
273   require(senderBalance >= amount, "BEP20: transfer amount exceeds balance");
274   _balances[sender] = senderBalance - amount;
275   _balances[recipient] += amount;
276
277   emit Transfer(sender, recipient, amount);
278
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 275

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
274   _balances[sender] = senderBalance - amount;
275   _balances[recipient] += amount;
276
277   emit Transfer(sender, recipient, amount);
278   }
279
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 404

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
403
404   uint256 public tokenLiquidityThreshold = 1e9 * 10**18; // 0.1%
405   uint256 public maxBuyLimit = 1e10 * 10**18; // 1%
406   uint256 public maxSellLimit = 1e10 * 10**18; // 1%
407   uint256 public maxWalletLimit = 1e10 * 10**18; // 1%
408
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 404

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
403
404   uint256 public tokenLiquidityThreshold = 1e9 * 10**18; // 0.1%
405   uint256 public maxBuyLimit = 1e10 * 10**18; // 1%
406   uint256 public maxSellLimit = 1e10 * 10**18; // 1%
407   uint256 public maxWalletLimit = 1e10 * 10**18; // 1%
408
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 405

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
404   uint256 public tokenLiquidityThreshold = 1e9 * 10**18; // 0.1%
405   uint256 public maxBuyLimit = 1e10 * 10**18; // 1%
406   uint256 public maxSellLimit = 1e10 * 10**18; // 1%
407   uint256 public maxWalletLimit = 1e10 * 10**18; // 1%
408
409
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 405

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
404   uint256 public tokenLiquidityThreshold = 1e9 * 10**18; // 0.1%
405   uint256 public maxBuyLimit = 1e10 * 10**18; // 1%
406   uint256 public maxSellLimit = 1e10 * 10**18; // 1%
407   uint256 public maxWalletLimit = 1e10 * 10**18; // 1%
408
409
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 406

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
405   uint256 public maxBuyLimit = 1e10 * 10**18; // 1%
406   uint256 public maxSellLimit = 1e10 * 10**18; // 1%
407   uint256 public maxWalletLimit = 1e10 * 10**18; // 1%
408
409   uint256 public genesis_block;
410
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 406

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
405   uint256 public maxBuyLimit = 1e10 * 10**18; // 1%
406   uint256 public maxSellLimit = 1e10 * 10**18; // 1%
407   uint256 public maxWalletLimit = 1e10 * 10**18; // 1%
408
409   uint256 public genesis_block;
410
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 407

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
406   uint256 public maxSellLimit = 1e10 * 10**18; // 1%
407   uint256 public maxWalletLimit = 1e10 * 10**18; // 1%
408
409   uint256 public genesis_block;
410   uint256 private deadline = 3;
411
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 407

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
406   uint256 public maxSellLimit = 1e10 * 10**18; // 1%
407   uint256 public maxWalletLimit = 1e10 * 10**18; // 1%
408
409   uint256 public genesis_block;
410   uint256 private deadline = 3;
411
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 440

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
439   constructor() BEP20("Sheikh Inu", "SHINU") {
440   _tokengeneration(msg.sender, 1e12 * 10**decimals());
441
442   IRouter _router = IRouter(0x10ED43C718714eb63d5aA57B78B54704E256024E);
443   // Create a pancake pair for this new token
444
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 440

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
439   constructor() BEP20("Sheikh Inu", "SHINU") {
440   _tokengeneration(msg.sender, 1e12 * 10**decimals());
441
442   IRouter _router = IRouter(0x10ED43C718714eb63d5aA57B78B54704E256024E);
443   // Create a pancake pair for this new token
444
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 472

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
471   require(currentAllowance >= amount, "BEP20: transfer amount exceeds allowance");
472   _approve(sender, _msgSender(), currentAllowance - amount);
473
474   return true;
475   }
476
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 482

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
481   {
482   _approve(_msgSender(), spender, _allowances[_msgSender()][spender] + addedValue);
483   return true;
484   }
485
486
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 493

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
492   require(currentAllowance >= subtractedValue, "BEP20: decreased allowance below
zero");
493   _approve(_msgSender(), spender, currentAllowance - subtractedValue);
494
495   return true;
496   }
497
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 516

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
515   require(amount <= maxBuyLimit, "You are exceeding maxBuyLimit");
516   require(balanceOf(recipient) + amount <= maxWalletLimit,"You are exceeding
maxWalletLimit");
517   }
518
519   if (sender != pair && !exemptFee[recipient] && !exemptFee[sender] && !_interlock) {
520
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 523

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
522   if (recipient != pair) {
523   require(balanceOf(recipient) + amount <= maxWalletLimit,"You are exceeding
maxWalletLimit");
524   }
525
526   if (coolDownEnabled) {
527
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 527

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
526   if (coolDownEnabled) {
527   uint256 timePassed = block.timestamp - _lastSell[sender];
528   require(timePassed >= coolDownTime, "Cooldown enabled");
529   _lastSell[sender] = block.timestamp;
530   }
531
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 540

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
539   !exemptFee[recipient] &&
540   block.number < genesis_block + deadline;
541
542   //set fee to zero if fees in contract are handled or exempted
543   if (_interlock || exemptFee[sender] || exemptFee[recipient])
544
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 549

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
548   feeswap =
549   sellTaxes.liquidity +
550   sellTaxes.marketing;
551   feesum = feeswap;
552   currentTaxes = sellTaxes;
553
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 555

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
554   feeswap =
555   taxes.liquidity +
556   taxes.marketing;
557   feesum = feeswap;
558   currentTaxes = taxes;
559
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 564

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
563
564   fee = (amount * feesum) / 100;
565
566   //send fees if threshold has been reached
567   //don't do this on buys, breaks swap
568
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 564

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
563
564   fee = (amount * feesum) / 100;
565
566   //send fees if threshold has been reached
567   //don't do this on buys, breaks swap
568
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 571

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
570   //rest to recipient
571   super._transfer(sender, recipient, amount - fee);
572   if (fee > 0) {
573   //send the fee to the contract
574   if (feeswap > 0) {
575
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 575

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
574   if (feeswap > 0) {
575   uint256 feeAmount = (amount * feeswap) / 100;
576   super._transfer(sender, address(this), feeAmount);
577   }
578
579
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 575

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
574   if (feeswap > 0) {
575   uint256 feeAmount = (amount * feeswap) / 100;
576   super._transfer(sender, address(this), feeAmount);
577   }
578
579
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 595

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
594   // Split the contract balance into halves
595   uint256 denominator = feeswap * 2;
596   uint256 tokensToAddLiquidityWith = (contractBalance * swapTaxes.liquidity) /
denominator;
597   uint256 toSwap = contractBalance - tokensToAddLiquidityWith;
598
599
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 596

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
595   uint256 denominator = feeswap * 2;
596   uint256 tokensToAddLiquidityWith = (contractBalance * swapTaxes.liquidity) /
denominator;
597   uint256 toSwap = contractBalance - tokensToAddLiquidityWith;
598
599   uint256 initialBalance = address(this).balance;
600
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 596

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
595   uint256 denominator = feeswap * 2;
596   uint256 tokensToAddLiquidityWith = (contractBalance * swapTaxes.liquidity) /
denominator;
597   uint256 toSwap = contractBalance - tokensToAddLiquidityWith;
598
599   uint256 initialBalance = address(this).balance;
600
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 597

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
596   uint256 tokensToAddLiquidityWith = (contractBalance * swapTaxes.liquidity) /
denominator;
597   uint256 toSwap = contractBalance - tokensToAddLiquidityWith;
598
599   uint256 initialBalance = address(this).balance;
600
601
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 603

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
602
603   uint256 deltaBalance = address(this).balance - initialBalance;
604   uint256 unitBalance = deltaBalance / (denominator - swapTaxes.liquidity);
605   uint256 ethToAddLiquidityWith = unitBalance * swapTaxes.liquidity;
606
607
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 604

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
603   uint256 deltaBalance = address(this).balance - initialBalance;
604   uint256 unitBalance = deltaBalance / (denominator - swapTaxes.liquidity);
605   uint256 ethToAddLiquidityWith = unitBalance * swapTaxes.liquidity;
606
607   if (ethToAddLiquidityWith > 0) {
608
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 604

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
603   uint256 deltaBalance = address(this).balance - initialBalance;
604   uint256 unitBalance = deltaBalance / (denominator - swapTaxes.liquidity);
605   uint256 ethToAddLiquidityWith = unitBalance * swapTaxes.liquidity;
606
607   if (ethToAddLiquidityWith > 0) {
608
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 605

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
604   uint256 unitBalance = deltaBalance / (denominator - swapTaxes.liquidity);
605   uint256 ethToAddLiquidityWith = unitBalance * swapTaxes.liquidity;
606
607   if (ethToAddLiquidityWith > 0) {
608   // Add liquidity to pancake
609
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 612

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
611
612   uint256 marketingAmt = unitBalance * 2 * swapTaxes.marketing;
613   if (marketingAmt > 0) {
614   payable(marketingWallet).sendValue(marketingAmt);
615   }
616
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 612

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
611
612   uint256 marketingAmt = unitBalance * 2 * swapTaxes.marketing;
613   if (marketingAmt > 0) {
614   payable(marketingWallet).sendValue(marketingAmt);
615   }
616
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 661

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
660   require(new_amount <= 1e10, "Swap threshold amount should be lower or equal to 1%
of tokens");
661   tokenLiquidityThreshold = new_amount * 10**decimals();
662   }
663
664   function SetBuyTaxes(
665
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 661

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
660   require(new_amount <= 1e10, "Swap threshold amount should be lower or equal to 1%
of tokens");
661   tokenLiquidityThreshold = new_amount * 10**decimals();
662   }
663
664   function SetBuyTaxes(
665
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 669

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
668   taxes = Taxes(_marketing, _liquidity);
669   require((_marketing + _liquidity) <= 10, "Must keep fees at 10% or less");
670   }
671
672   function SetSellTaxes(
673
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 677

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
676   sellTaxes = Taxes(_marketing, _liquidity);
677   require((_marketing + _liquidity) <= 10, "Must keep fees at 10% or less");
678   }
679
680   function updateRouterAndPair(address newRouter, address newPair) external onlyOwner
{
681
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 704

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
703   function updateCooldown(bool state, uint256 time) external onlyOwner {
704   coolDownTime = time * 1 seconds;
705   coolDownEnabled = state;
706   require(time <= 300, "cooldown timer cannot exceed 5 minutes");
707   }
708
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 714

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
713   function bulkExemptFee(address[] memory accounts, bool state) external onlyOwner {
714   for (uint256 i = 0; i < accounts.length; i++) {
715   exemptFee[accounts[i]] = state;
716   }
717   }
718
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 723

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
722   require(maxWallet >= 1e10, "Cannot set max wallet amount lower than 1%");
723   maxBuyLimit = maxBuy * 10**decimals();
724   maxSellLimit = maxSell * 10**decimals();
725   maxWalletLimit = maxWallet * 10**decimals();
726   }
727
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 723

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
722   require(maxWallet >= 1e10, "Cannot set max wallet amount lower than 1%");
723   maxBuyLimit = maxBuy * 10**decimals();
724   maxSellLimit = maxSell * 10**decimals();
725   maxWalletLimit = maxWallet * 10**decimals();
726   }
727
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 724

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
723   maxBuyLimit = maxBuy * 10**decimals();
724   maxSellLimit = maxSell * 10**decimals();
725   maxWalletLimit = maxWallet * 10**decimals();
726   }
727
728
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 724

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
723   maxBuyLimit = maxBuy * 10**decimals();
724   maxSellLimit = maxSell * 10**decimals();
725   maxWalletLimit = maxWallet * 10**decimals();
726   }
727
728
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 725

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
724   maxSellLimit = maxSell * 10**decimals();
725   maxWalletLimit = maxWallet * 10**decimals();
726   }
727
728   function rescueBNB(uint256 weiAmount) external onlyOwner {
729
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 725

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
724   maxSellLimit = maxSell * 10**decimals();
725   maxWalletLimit = maxWallet * 10**decimals();
726   }
727
728   function rescueBNB(uint256 weiAmount) external onlyOwner {
729
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 11

low SEVERITY

The current pragma Solidity directive is ""^0.8.8"". 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

- SheikhInu.sol

Locations

```
10
11   pragma solidity ^0.8.8;
12
13   abstract contract Context {
14   function _msgSender() internal view virtual returns (address) {
15
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 623

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
622   address[] memory path = new address[](2);
623   path[0] = address(this);
624   path[1] = router.WETH();
625
626   _approve(address(this), address(router), tokenAmount);
627
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 624

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
623   path[0] = address(this);
624   path[1] = router.WETH();
625
626   _approve(address(this), address(router), tokenAmount);
627
628
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 715

low SEVERITY

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

Source File

- SheikhInu.sol

Locations

```
714   for (uint256 i = 0; i < accounts.length; i++) {
715   exemptFee[accounts[i]] = state;
716   }
717   }
718
719
```

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

LINE 540

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

- SheikhInu.sol

Locations

```
539   !exemptFee[recipient] &&
540   block.number < genesis_block + deadline;
541
542   //set fee to zero if fees in contract are handled or exempted
543   if (_interlock || exemptFee[sender] || exemptFee[recipient])
544
```

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

LINE 689

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

- SheikhInu.sol

Locations

```
688   providingLiquidity = true;
689   genesis_block = block.number;
690   }
691
692   function updatedeadline(uint256 _deadline) external onlyOwner {
693
```

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.