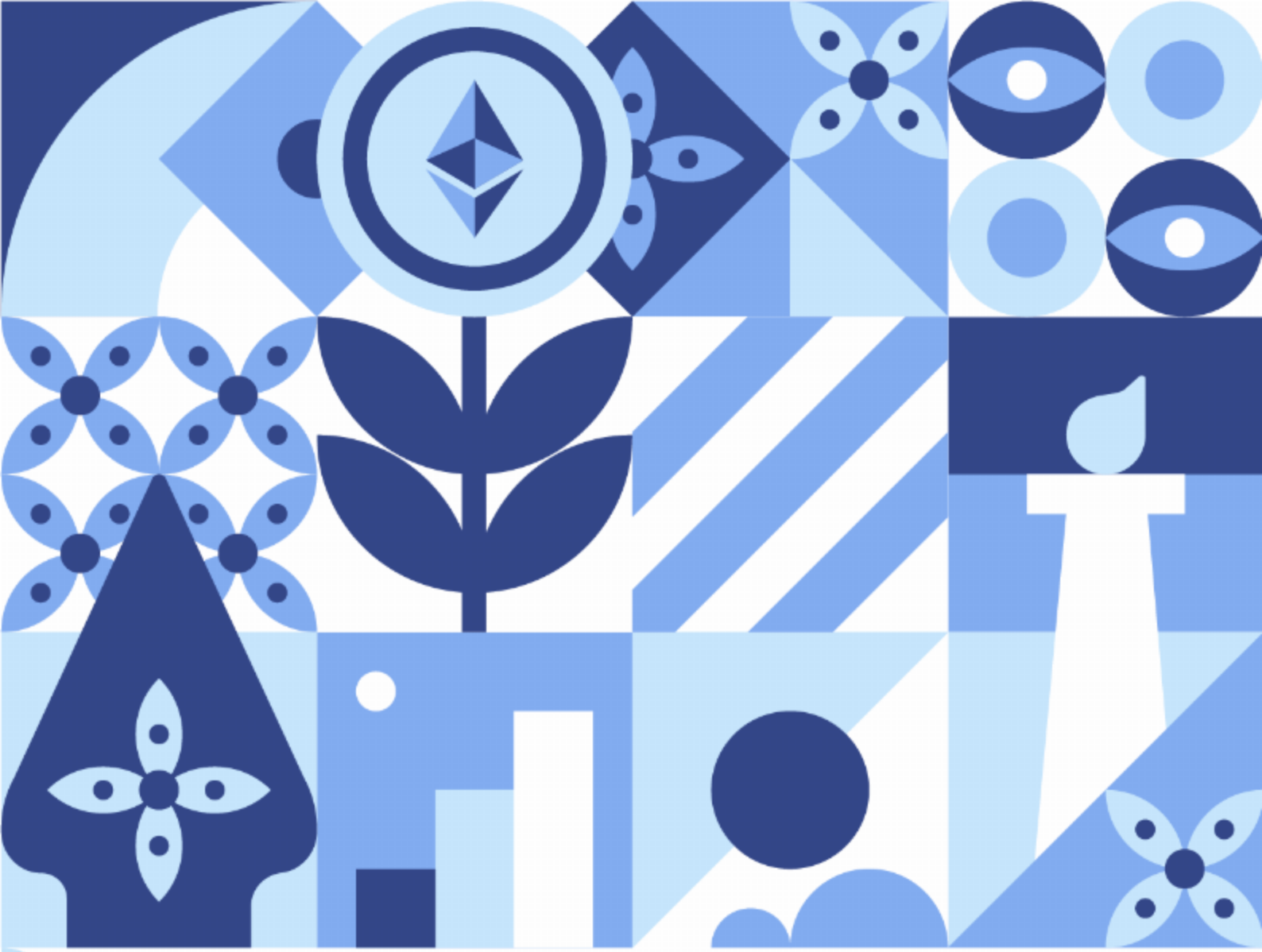

SAUDI SHIBA INU

Smart Contract Audit Report

SYSFIXED

01 Aug 2022

TABLE OF CONTENTS

Audited Details

- Audited Project
- Blockchain
- Addresses
- Project Website
- Codebase

Summary

- Contract Summary
- Audit Findings Summary
- Vulnerabilities Summary

Conclusion

Audit Results

Smart Contract Analysis

- Detected Vulnerabilities

Disclaimer

About Us

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
SAUDI SHIBA INU	SAUDISHIB	Ethereum

Addresses

Contract address	0x34d31446a522252270b89b09016296ec4c98e23d
Contract deployer address	0x95573FB29D9E17fF77d0bAEFBA46A2077eFe6f47

Project Website

https://saudishibatoken.com/

Codebase

https://etherscan.io/address/0x34d31446a522252270b89b09016296ec4c98e23d#code

SUMMARY

The Saudis will buy tons of Bitcoin and Shiba Inu and now the official Saudi version of Shiba Inu is born. Be ready for the biggest pump to the moon!

Contract Summary

Documentation Quality

SAUDI SHIBA 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 SAUDI SHIBA 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 969.
- SWC-101 | It is recommended to use vetted safe math libraries for arithmetic operations consistently on lines 120, 156, 179, 180, 219, 259, 531, 941, 941, 941, 941, 942, 942, 972, 972, 972, 972, 973, 973, 973, 973, 974, 974, 974, 974, 1203, 1206, 1227, 1229, 1276, 1283, 1346, 1367, 1375, 1432, 1206 and 1229.
- SWC-103 | Pragma statements can be allowed to float when a contract is intended on lines 10.
- 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 1204, 1205, 1205, 1228, 1229, 1229, 1348, 1349, 1351, 1352, 1507 and 1508.
- SWC-115 | tx.origin should not be used for authorization, use msg.sender instead on lines 1426.

CONCLUSION

We have audited the SAUDI SHIBA INU project released on July 2022 to discover issues and identify potential security vulnerabilities in SAUDI SHIBA 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 SAUDI SHIBA 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 is set, a state variable visibility is not set, tx.origin as a part of authorization control 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.

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.	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.	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.	PASS
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	Sunday Jul 31 2022 09:56:56 GMT+0000 (Coordinated Universal Time)
Finished	Monday Aug 01 2022 17:37:20 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	SaudiShibaInu.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	COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED	low	acknowledged
SWC-101	COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	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-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 120

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
119   function add(uint256 a, uint256 b) internal pure returns (uint256) {
120   uint256 c = a + b;
121   require(c >= a, "SafeMath: addition overflow");
122
123   return c;
124
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 156

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
155   require(b <= a, errorMessage);
156   uint256 c = a - b;
157
158   return c;
159   }
160
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 179

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
178
179   uint256 c = a * b;
180   require(c / a == b, "SafeMath: multiplication overflow");
181
182   return c;
183
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 180

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
179   uint256 c = a * b;
180   require(c / a == b, "SafeMath: multiplication overflow");
181
182   return c;
183   }
184
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 219

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
218   require(b > 0, errorMessage);
219   uint256 c = a / b;
220   // assert(a == b * c + a % b); // There is no case in which this doesn't hold
221
222   return c;
223
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 259

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
258   require(b != 0, errorMessage);
259   return a % b;
260   }
261   }
262
263
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 531

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
530   _owner = address(0);
531   _lockTime = block.timestamp + time;
532   emit OwnershipTransferred(_owner, address(0));
533   }
534
535
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 941

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
940   uint256 private constant MAX = ~uint256(0);
941   uint256 private _tTotal = 1000000000 * 10**6 * 10**8;
942   uint256 private _rTotal = (MAX - (MAX % _tTotal));
943   uint256 private _tFeeTotal;
944
945
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 941

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
940   uint256 private constant MAX = ~uint256(0);
941   uint256 private _tTotal = 1000000000 * 10**6 * 10**8;
942   uint256 private _rTotal = (MAX - (MAX % _tTotal));
943   uint256 private _tFeeTotal;
944
945
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 941

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
940   uint256 private constant MAX = ~uint256(0);
941   uint256 private _tTotal = 1000000000 * 10**6 * 10**8;
942   uint256 private _rTotal = (MAX - (MAX % _tTotal));
943   uint256 private _tFeeTotal;
944
945
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 941

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
940   uint256 private constant MAX = ~uint256(0);
941   uint256 private _tTotal = 1000000000 * 10**6 * 10**8;
942   uint256 private _rTotal = (MAX - (MAX % _tTotal));
943   uint256 private _tFeeTotal;
944
945
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 942

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
941   uint256 private _tTotal = 1000000000 * 10**6 * 10**8;
942   uint256 private _rTotal = (MAX - (MAX % _tTotal));
943   uint256 private _tFeeTotal;
944
945
946
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 942

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
941   uint256 private _tTotal = 1000000000 * 10**6 * 10**8;
942   uint256 private _rTotal = (MAX - (MAX % _tTotal));
943   uint256 private _tFeeTotal;
944
945
946
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 972

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
971
972   uint256 public _maxTxAmount = 1000000000 * 10**6 * 10**8;
973   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**8;
974   uint256 public _maxWalletSize = 1 * 10**13 * 10**8;
975
976
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 972

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
971
972   uint256 public _maxTxAmount = 1000000000 * 10**6 * 10**8;
973   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**8;
974   uint256 public _maxWalletSize = 1 * 10**13 * 10**8;
975
976
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 972

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
971
972   uint256 public _maxTxAmount = 1000000000 * 10**6 * 10**8;
973   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**8;
974   uint256 public _maxWalletSize = 1 * 10**13 * 10**8;
975
976
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 972

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
971
972   uint256 public _maxTxAmount = 1000000000 * 10**6 * 10**8;
973   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**8;
974   uint256 public _maxWalletSize = 1 * 10**13 * 10**8;
975
976
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 973

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
972   uint256 public _maxTxAmount = 1000000000 * 10**6 * 10**8;
973   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**8;
974   uint256 public _maxWalletSize = 1 * 10**13 * 10**8;
975
976   event botAddedToBlacklist(address account);
977
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 973

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
972   uint256 public _maxTxAmount = 1000000000 * 10**6 * 10**8;
973   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**8;
974   uint256 public _maxWalletSize = 1 * 10**13 * 10**8;
975
976   event botAddedToBlacklist(address account);
977
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 973

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
972   uint256 public _maxTxAmount = 1000000000 * 10**6 * 10**8;
973   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**8;
974   uint256 public _maxWalletSize = 1 * 10**13 * 10**8;
975
976   event botAddedToBlacklist(address account);
977
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 973

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
972   uint256 public _maxTxAmount = 1000000000 * 10**6 * 10**8;
973   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**8;
974   uint256 public _maxWalletSize = 1 * 10**13 * 10**8;
975
976   event botAddedToBlacklist(address account);
977
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 974

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
973   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**8;
974   uint256 public _maxWalletSize = 1 * 10**13 * 10**8;
975
976   event botAddedToBlacklist(address account);
977   event botRemovedFromBlacklist(address account);
978
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 974

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
973   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**8;
974   uint256 public _maxWalletSize = 1 * 10**13 * 10**8;
975
976   event botAddedToBlacklist(address account);
977   event botRemovedFromBlacklist(address account);
978
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 974

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
973   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**8;
974   uint256 public _maxWalletSize = 1 * 10**13 * 10**8;
975
976   event botAddedToBlacklist(address account);
977   event botRemovedFromBlacklist(address account);
978
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 974

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
973   uint256 private numTokensSellToAddToLiquidity = 500000 * 10**6 * 10**8;
974   uint256 public _maxWalletSize = 1 * 10**13 * 10**8;
975
976   event botAddedToBlacklist(address account);
977   event botRemovedFromBlacklist(address account);
978
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 1203

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1202   require(_isBlackListedBot[account], "Account is not blacklisted");
1203   for (uint256 i = 0; i < _blackListedBots.length; i++) {
1204   if (_blackListedBots[i] == account) {
1205   _blackListedBots[i] = _blackListedBots[
1206   _blackListedBots.length - 1
1207
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1206

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1205   _blackListedBots[i] = _blackListedBots[
1206   _blackListedBots.length - 1
1207   ];
1208   _isBlackListedBot[account] = false;
1209   _blackListedBots.pop();
1210
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 1227

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1226   require(_isExcluded[account], "Account is not excluded");
1227   for (uint256 i = 0; i < _excluded.length; i++) {
1228   if (_excluded[i] == account) {
1229   _excluded[i] = _excluded[_excluded.length - 1];
1230   _tOwned[account] = 0;
1231
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1229

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1228   if (_excluded[i] == account) {
1229   _excluded[i] = _excluded[_excluded.length - 1];
1230   _tOwned[account] = 0;
1231   _isExcluded[account] = false;
1232   _excluded.pop();
1233
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1276

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1275   function setMaxTxPercent(uint256 maxTxPercent) external onlyOwner {
1276   _maxTxAmount = _tTotal.mul(maxTxPercent).div(10**2);
1277   }
1278
1279   function _setMaxWalletSizePercent(uint256 maxWalletSize)
1280
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1283

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1282   {
1283   _maxWalletSize = _tTotal.mul(maxWalletSize).div(10**2);
1284   }
1285
1286   function setSwapAndLiquifyEnabled(bool _enabled) public onlyOwner {
1287
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 1346

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1345   uint256 tSupply = _tTotal;
1346   for (uint256 i = 0; i < _excluded.length; i++) {
1347   if (
1348   _rOwned[_excluded[i]] > rSupply ||
1349   _tOwned[_excluded[i]] > tSupply
1350
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1367

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1366   function calculateTaxFee(uint256 _amount) private view returns (uint256) {
1367   return _amount.mul(_taxFee).div(10**2);
1368   }
1369
1370   function calculateLiquidityFee(uint256 _amount)
1371
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1375

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1374   {
1375   return _amount.mul(_liquidityFee).div(10**2);
1376   }
1377
1378   function removeAllFee() private {
1379
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1432

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1431   if(to != uniswapV2Pair) {
1432   require(balanceOf(to) + amount < _maxWalletSize, "TOKEN: Balance exceeds wallet
size!");
1433   }
1434   }
1435
1436
```

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

LINE 1206

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1205   _blackListedBots[i] = _blackListedBots[
1206   _blackListedBots.length - 1
1207   ];
1208   _isBlackListedBot[account] = false;
1209   _blackListedBots.pop();
1210
```

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

LINE 1229

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1228   if (_excluded[i] == account) {
1229   _excluded[i] = _excluded[_excluded.length - 1];
1230   _tOwned[account] = 0;
1231   _isExcluded[account] = false;
1232   _excluded.pop();
1233
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 10

low SEVERITY

The current pragma Solidity directive is ""^0.8.10"". 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

- SaudiShibaInu.sol

Locations

```
9
10   pragma solidity ^0.8.10;
11
12   // SPDX-License-Identifier: Unlicensed
13   interface IERC20 {
14
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 969

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
968
969   bool inSwapAndLiquify;
970   bool public swapAndLiquifyEnabled = true;
971
972   uint256 public _maxTxAmount = 1000000000 * 10**6 * 10**8;
973
```

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

LINE 1426

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

- SaudiShibaInu.sol

Locations

```
1425   require(!_isBlackListedBot[msg.sender], "you are blacklisted");
1426   require(!_isBlackListedBot[tx.origin], "blacklisted");
1427
1428   if (!_isExcludedFromLimit[from] && !_isExcludedFromLimit[to]) {
1429   require(amount <= _maxTxAmount,"Transfer amount exceeds the maxTxAmount.");
1430
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1204

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1203   for (uint256 i = 0; i < _blackListedBots.length; i++) {
1204   if (_blackListedBots[i] == account) {
1205   _blackListedBots[i] = _blackListedBots[
1206   _blackListedBots.length - 1
1207   ];
1208
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1205

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1204   if (_blackListedBots[i] == account) {
1205   _blackListedBots[i] = _blackListedBots[
1206   _blackListedBots.length - 1
1207   ];
1208   _isBlackListedBot[account] = false;
1209
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1205

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1204   if (_blackListedBots[i] == account) {
1205   _blackListedBots[i] = _blackListedBots[
1206   _blackListedBots.length - 1
1207   ];
1208   _isBlackListedBot[account] = false;
1209
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1228

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1227   for (uint256 i = 0; i < _excluded.length; i++) {
1228   if (_excluded[i] == account) {
1229   _excluded[i] = _excluded[_excluded.length - 1];
1230   _tOwned[account] = 0;
1231   _isExcluded[account] = false;
1232
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1229

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1228   if (_excluded[i] == account) {
1229   _excluded[i] = _excluded[_excluded.length - 1];
1230   _tOwned[account] = 0;
1231   _isExcluded[account] = false;
1232   _excluded.pop();
1233
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1229

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1228   if (_excluded[i] == account) {
1229   _excluded[i] = _excluded[_excluded.length - 1];
1230   _tOwned[account] = 0;
1231   _isExcluded[account] = false;
1232   _excluded.pop();
1233
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1348

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1347   if (
1348   _rOwned[_excluded[i]] > rSupply ||
1349   _tOwned[_excluded[i]] > tSupply
1350   ) return (_rTotal, _tTotal);
1351   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
1352
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1349

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1348   _rOwned[_excluded[i]] > rSupply ||
1349   _tOwned[_excluded[i]] > tSupply
1350   ) return (_rTotal, _tTotal);
1351   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
1352   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
1353
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1351

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1350   ) return (_rTotal, _tTotal);
1351   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
1352   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
1353   }
1354   if (rSupply < _rTotal.div(_tTotal)) return (_rTotal, _tTotal);
1355
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1352

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1351   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
1352   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
1353   }
1354   if (rSupply < _rTotal.div(_tTotal)) return (_rTotal, _tTotal);
1355   return (rSupply, tSupply);
1356
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1507

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1506   address[] memory path = new address[](2);
1507   path[0] = address(this);
1508   path[1] = uniswapV2Router.WETH();
1509
1510   _approve(address(this), address(uniswapV2Router), tokenAmount);
1511
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1508

low SEVERITY

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

Source File

- SaudiShibaInu.sol

Locations

```
1507   path[0] = address(this);
1508   path[1] = uniswapV2Router.WETH();
1509
1510   _approve(address(this), address(uniswapV2Router), tokenAmount);
1511
1512
```

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.