Strelka Coin

Smart Contract Audit Report

SYSFIXED

01 Feb 2023

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
Strelka Coin	STRELKA	Binance Smart Chain

Addresses

Contract address	0x5A21A450b39dD19108deD9F70EC1285A757efe25
Contract deployer address	0x0A082068Fa05253eAB2EaeD698edA443C57452c2

Project Website

https://www.strelkaproject.com/

Codebase

https://bscscan.com/address/0x5A21A450b39dD19108deD9F70EC1285A757efe25#code

SUMMARY

Strelka Coin is an innovative project combining a revolutionary NFT P2E Game with charity partnerships with the biggest animal welfare associations in the world. The NFT P2E Game is ready and playable at launch. Charity partnership with Humane Society International - the biggest animal welfare association in the world. Legal entity and an experienced team. Product (NFT Play to Earn Game) ready and playable at launch. Partnerships with the biggest BSC influencers. Partnerships with celebrities

Contract Summary

Documentation Quality

Strelka Coin 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 Strelka Coin 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 328, 329, 341 and 342.
- SWC-101 | It is recommended to use vetted safe math libraries for arithmetic operations consistently on lines 215, 215, 215, 215, 216, 216, 217, 217, 218, 218, 219, 219, 220, 220, 251, 252, 256, 287, 326, 326, 343, 343, 408, 423, 425, 429, 431, 446, 463, 464, 470, 470, 470, 470, 471, 471, 472, 474, 486, 486, 486, 486, 487, 527 and 527.
- 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 236 and 237.

CONCLUSION

We have audited the NamaFile project released on January 2023 to discover issues and identify potential security vulnerabilities in NamaFile 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 NamaFile 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 state variable visibility is not set, 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 and it's best practice to set the visibility of state variables explicitly. The default visibility for "_balances" is internal. Other possible visibility settings are public and private.

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.	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.	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	Tuesday Jan 31 2023 13:07:38 GMT+0000 (Coordinated Universal Time)
Finished	Wednesday Feb 01 2023 03:32:33 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	Strelka.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-108	STATE VARIABLE VISIBILITY IS NOT SET.	low	acknowledged
SWC-108	STATE VARIABLE VISIBILITY IS NOT SET.	low	acknowledged
SWC-108	STATE VARIABLE VISIBILITY IS NOT SET.	low	acknowledged
SWC-108	STATE VARIABLE VISIBILITY IS NOT SET.	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 215

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
214   lastDistributeTime = block.timestamp;
215   uint256 totalTax = taxToReward + taxToLiquidity + taxToOperations + taxToMarketing
+ taxToCharity;
216   uint256 amountForReward = amount * taxToReward / totalTax;
217   uint256 amountForLiquidity = amount*taxToLiquidity / totalTax;
218   uint256 amountForOperations = amount*taxToOperations / totalTax;
219
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 215

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
214   lastDistributeTime = block.timestamp;
215   uint256 totalTax = taxToReward + taxToLiquidity + taxToOperations + taxToMarketing
+ taxToCharity;
216   uint256 amountForReward = amount * taxToReward / totalTax;
217   uint256 amountForLiquidity = amount*taxToLiquidity / totalTax;
218   uint256 amountForOperations = amount*taxToOperations / totalTax;
219
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 215

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
214   lastDistributeTime = block.timestamp;
215   uint256 totalTax = taxToReward + taxToLiquidity + taxToOperations + taxToMarketing
+ taxToCharity;
216   uint256 amountForReward = amount * taxToReward / totalTax;
217   uint256 amountForLiquidity = amount*taxToLiquidity / totalTax;
218   uint256 amountForOperations = amount*taxToOperations / totalTax;
219
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 215

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
214   lastDistributeTime = block.timestamp;
215   uint256 totalTax = taxToReward + taxToLiquidity + taxToOperations + taxToMarketing
+ taxToCharity;
216   uint256 amountForReward = amount * taxToReward / totalTax;
217   uint256 amountForLiquidity = amount*taxToLiquidity / totalTax;
218   uint256 amountForOperations = amount*taxToOperations / totalTax;
219
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 216

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
215   uint256 totalTax = taxToReward + taxToLiquidity + taxToOperations + taxToMarketing
+ taxToCharity;
216   uint256 amountForReward = amount * taxToReward / totalTax;
217   uint256 amountForLiquidity = amount*taxToLiquidity / totalTax;
218   uint256 amountForOperations = amount*taxToOperations / totalTax;
219   uint256 amountForMarketing = amount*taxToMarketing / totalTax;
220
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 216

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
215   uint256 totalTax = taxToReward + taxToLiquidity + taxToOperations + taxToMarketing
+ taxToCharity;
216   uint256 amountForReward = amount * taxToReward / totalTax;
217   uint256 amountForLiquidity = amount*taxToLiquidity / totalTax;
218   uint256 amountForOperations = amount*taxToOperations / totalTax;
219   uint256 amountForMarketing = amount*taxToMarketing / totalTax;
220
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 217

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
216   uint256 amountForReward = amount * taxToReward / totalTax;
217   uint256 amountForLiquidity = amount*taxToLiquidity / totalTax;
218   uint256 amountForOperations = amount*taxToOperations / totalTax;
219   uint256 amountForMarketing = amount*taxToMarketing / totalTax;
220   uint256 amountForCharity = amount*taxToCharity / totalTax;
221
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 217

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
216   uint256 amountForReward = amount * taxToReward / totalTax;
217   uint256 amountForLiquidity = amount*taxToLiquidity / totalTax;
218   uint256 amountForOperations = amount*taxToOperations / totalTax;
219   uint256 amountForMarketing = amount*taxToMarketing / totalTax;
220   uint256 amountForCharity = amount*taxToCharity / totalTax;
221
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 218

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
217   uint256 amountForLiquidity = amount*taxToLiquidity / totalTax;
218   uint256 amountForOperations = amount*taxToOperations / totalTax;
219   uint256 amountForMarketing = amount*taxToMarketing / totalTax;
220   uint256 amountForCharity = amount*taxToCharity / totalTax;
221
222
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 218

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
217   uint256 amountForLiquidity = amount*taxToLiquidity / totalTax;
218   uint256 amountForOperations = amount*taxToOperations / totalTax;
219   uint256 amountForMarketing = amount*taxToMarketing / totalTax;
220   uint256 amountForCharity = amount*taxToCharity / totalTax;
221
222
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 219

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
218   uint256 amountForOperations = amount*taxToOperations / totalTax;
219   uint256 amountForMarketing = amount*taxToMarketing / totalTax;
220   uint256 amountForCharity = amount*taxToCharity / totalTax;
221
222   token.transfer(rewardPool, amountForReward);
223
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 219

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
218   uint256 amountForOperations = amount*taxToOperations / totalTax;
219   uint256 amountForMarketing = amount*taxToMarketing / totalTax;
220   uint256 amountForCharity = amount*taxToCharity / totalTax;
221
222   token.transfer(rewardPool, amountForReward);
223
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 220

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
219   uint256 amountForMarketing = amount*taxToMarketing / totalTax;
220   uint256 amountForCharity = amount*taxToCharity / totalTax;
221
222   token.transfer(rewardPool, amountForReward);
223
224
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 220

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
219   uint256 amountForMarketing = amount*taxToMarketing / totalTax;
220   uint256 amountForCharity = amount*taxToCharity / totalTax;
221
222   token.transfer(rewardPool, amountForReward);
223
224
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 251

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
250   ) internal {
251   uint256 halfTokenAmount = tokenAmount / 2;
252   uint256 busdAmount = tokenAmount - halfTokenAmount;
253   uint256 busdBalanceBefore = IERC20(BUSD).balanceOf(address(this));
254   swapTokens(halfTokenAmount, address(this));
255
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 252

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
251   uint256 halfTokenAmount = tokenAmount / 2;
252   uint256 busdAmount = tokenAmount - halfTokenAmount;
253   uint256 busdBalanceBefore = IERC20(BUSD).balanceOf(address(this));
254   swapTokens(halfTokenAmount, address(this));
255   uint256 busdBalanceAfter = IERC20(BUSD).balanceOf(address(this));
256
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 256

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
255   uint256 busdBalanceAfter = IERC20(BUSD).balanceOf(address(this));
256   uint256 busdBalance = busdBalanceAfter - busdBalanceBefore;
257   IERC20(BUSD).approve(address(router), busdBalance);
258   token.approve(address(router), halfTokenAmount);
259   router.addLiquidity(
260
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 287

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
286   function shouldDistribute() internal view returns(bool) {
287   return lastDistributeTime + minPeriod < block.timestamp;
288   }
289   function setMinPeriod(uint256 _minPeriod) external override authorized {
290   minPeriod = _minPeriod;
291
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 326

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
325
326   uint256 constant _totalSupply = 100000000 * (10 ** _decimals);
327
328   mapping (address => uint256) _balances;
329   mapping (address => mapping (address => uint256)) _allowances;
330
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 326

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
325
326   uint256 constant _totalSupply = 100000000 * (10 ** _decimals);
327
328   mapping (address => uint256) _balances;
329   mapping (address => mapping (address => uint256)) _allowances;
330
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 343

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
342   mapping (address => bool) isTxLimitExempt;
343   uint256 public _maxTxAmount = 4500 * (10 ** _decimals);
344   bool public antiBotEnabled = true;
345   uint256 public cooldownTime = 30 seconds;
346   mapping(address => uint256) public purchasedTime;
347
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 343

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
342   mapping (address => bool) isTxLimitExempt;
343   uint256 public _maxTxAmount = 4500 * (10 ** _decimals);
344   bool public antiBotEnabled = true;
345   uint256 public cooldownTime = 30 seconds;
346   mapping(address => uint256) public purchasedTime;
347
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 408

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
407   if(_allowances[sender][msg.sender] != _totalSupply) {
408   _allowances[sender][msg.sender] = _allowances[sender][msg.sender] - amount;
409   }
410   return _transferFrom(sender, recipient, amount);
411   }
412
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 423

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
422   purchasedTime[recipient] = block.timestamp;
423   _balances[sender] = _balances[sender] - amount;
424   uint256 amountReceived = takeTax(sender, amount);
425   _balances[recipient] = _balances[recipient] + amountReceived;
426   emit Transfer(sender, recipient, amountReceived);
427
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 425

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
424   uint256 amountReceived = takeTax(sender, amount);
425   _balances[recipient] = _balances[recipient] + amountReceived;
426   emit Transfer(sender, recipient, amountReceived);
427   return true;
428   } else if(!addingLiquidity && recipient==address(pair)) {
429
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 429

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
428   } else if(!addingLiquidity && recipient==address(pair)) {
429   _balances[sender] = _balances[sender] - amount;
430   uint256 amountReceived = takeTax(sender, amount);
431   _balances[recipient] = _balances[recipient] + amountReceived;
432   emit Transfer(sender, recipient, amountReceived);
433
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 431

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
430   uint256 amountReceived = takeTax(sender, amount);
431   _balances[recipient] = _balances[recipient] + amountReceived;
432   emit Transfer(sender, recipient, amountReceived);
433   return true;
434   } else {
435
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 446

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
445   require(amount <= _maxTxAmount || isTxLimitExempt[sender], "TX Limit Exceeded");
446   require(block.timestamp>purchasedTime[recipient]+cooldownTime, "You can make
another purchase after cooldown time");
447   }
448
449   function setIsTxLimitExempt(address holder, bool exempt) external onlyOwner {
450
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 463

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
462   function _basicTransfer(address sender, address recipient, uint256 amount) internal
returns (bool) {
463   _balances[sender] = _balances[sender] - amount;
464   _balances[recipient] = _balances[recipient] + amount;
465   emit Transfer(sender, recipient, amount);
466   return true;
467
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 464

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
463   _balances[sender] = _balances[sender] - amount;
464   _balances[recipient] = _balances[recipient] + amount;
465   emit Transfer(sender, recipient, amount);
466   return true;
467   }
468
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 470

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
469   function takeTax(address sender, uint256 amount) internal returns (uint256) {
470   uint256 totalTax = taxRewardPool + taxLiquidity + taxOperations + taxMarketing +
taxCharity;
471   uint256 buyTaxAmount = amount*(totalTax)/feeDenominator;
472   _balances[address(taxHandler)] = _balances[address(taxHandler)] + buyTaxAmount;
473   emit Transfer(sender, address(taxHandler), buyTaxAmount);
474
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 470

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
469   function takeTax(address sender, uint256 amount) internal returns (uint256) {
470   uint256 totalTax = taxRewardPool + taxLiquidity + taxOperations + taxMarketing +
taxCharity;
471   uint256 buyTaxAmount = amount*(totalTax)/feeDenominator;
472   _balances[address(taxHandler)] = _balances[address(taxHandler)] + buyTaxAmount;
473   emit Transfer(sender, address(taxHandler), buyTaxAmount);
474
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 470

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
469   function takeTax(address sender, uint256 amount) internal returns (uint256) {
470   uint256 totalTax = taxRewardPool + taxLiquidity + taxOperations + taxMarketing +
taxCharity;
471   uint256 buyTaxAmount = amount*(totalTax)/feeDenominator;
472   _balances[address(taxHandler)] = _balances[address(taxHandler)] + buyTaxAmount;
473   emit Transfer(sender, address(taxHandler), buyTaxAmount);
474
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 470

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
469   function takeTax(address sender, uint256 amount) internal returns (uint256) {
470   uint256 totalTax = taxRewardPool + taxLiquidity + taxOperations + taxMarketing +
taxCharity;
471   uint256 buyTaxAmount = amount*(totalTax)/feeDenominator;
472   _balances[address(taxHandler)] = _balances[address(taxHandler)] + buyTaxAmount;
473   emit Transfer(sender, address(taxHandler), buyTaxAmount);
474
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 471

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
470   uint256 totalTax = taxRewardPool + taxLiquidity + taxOperations + taxMarketing +
taxCharity;
471   uint256 buyTaxAmount = amount*(totalTax)/feeDenominator;
472   _balances[address(taxHandler)] = _balances[address(taxHandler)] + buyTaxAmount;
473   emit Transfer(sender, address(taxHandler), buyTaxAmount);
474   return amount - buyTaxAmount;
475
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 471

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
470   uint256 totalTax = taxRewardPool + taxLiquidity + taxOperations + taxMarketing +
taxCharity;
471   uint256 buyTaxAmount = amount*(totalTax)/feeDenominator;
472   _balances[address(taxHandler)] = _balances[address(taxHandler)] + buyTaxAmount;
473   emit Transfer(sender, address(taxHandler), buyTaxAmount);
474   return amount - buyTaxAmount;
475
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 472

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
471   uint256 buyTaxAmount = amount*(totalTax)/feeDenominator;
472   _balances[address(taxHandler)] = _balances[address(taxHandler)] + buyTaxAmount;
473   emit Transfer(sender, address(taxHandler), buyTaxAmount);
474   return amount - buyTaxAmount;
475   }
476
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 474

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
473   emit Transfer(sender, address(taxHandler), buyTaxAmount);
474   return amount - buyTaxAmount;
475   }
476
477   function setTaxs(
478
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 486

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
485   require(_feeDenominator<=10000, "Fee denominator can not be set over 100%");
486   uint256 _total = _reward + _liquidity + _operations + _marketing + _charity;
487   require(_total<=_feeDenominator/10, "Total tax can not be set over 10%"); /// Tax
cannot exceed 10%
488
489   taxRewardPool = _reward;
490
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 486

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
485   require(_feeDenominator<=10000, "Fee denominator can not be set over 100%");
486   uint256 _total = _reward + _liquidity + _operations + _marketing + _charity;
487   require(_total<=_feeDenominator/10, "Total tax can not be set over 10%"); /// Tax
cannot exceed 10%
488
489   taxRewardPool = _reward;
490
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 486

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
485   require(_feeDenominator<=10000, "Fee denominator can not be set over 100%");
486   uint256 _total = _reward + _liquidity + _operations + _marketing + _charity;
487   require(_total<=_feeDenominator/10, "Total tax can not be set over 10%"); /// Tax
cannot exceed 10%
488
489   taxRewardPool = _reward;
490
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 486

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
485   require(_feeDenominator<=10000, "Fee denominator can not be set over 100%");
486   uint256 _total = _reward + _liquidity + _operations + _marketing + _charity;
487   require(_total<=_feeDenominator/10, "Total tax can not be set over 10%"); /// Tax
cannot exceed 10%
488
489   taxRewardPool = _reward;
490
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 487

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
486   uint256 _total = _reward + _liquidity + _operations + _marketing + _charity;
487   require(_total<=_feeDenominator/10, "Total tax can not be set over 10%"); /// Tax
cannot exceed 10%
488
489   taxRewardPool = _reward;
490   taxLiquidity = _liquidity;
491
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 527

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
526   require(amount > 4000);
527   _maxTxAmount = amount * (10**_decimals);
528   }
529   function setAntibot(bool _enable) external onlyOwner {
530   antiBotEnabled = _enable;
531
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 527

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
526   require(amount > 4000);
527   _maxTxAmount = amount * (10**_decimals);
528   }
529   function setAntibot(bool _enable) external onlyOwner {
530   antiBotEnabled = _enable;
531
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 328

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
327
328   mapping (address => uint256) _balances;
329   mapping (address => mapping (address => uint256)) _allowances;
330
331   uint256 public taxRewardPool = 100;
332
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 329

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
328   mapping (address => uint256) _balances;
329   mapping (address => mapping (address => uint256)) _allowances;
330
331   uint256 public taxRewardPool = 100;
332   uint256 public taxLiquidity = 100;
333
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 341

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
340   bool public addingLiquidity;
341   bool processing = false;
342   mapping (address => bool) isTxLimitExempt;
343   uint256 public _maxTxAmount = 4500 * (10 ** _decimals);
344   bool public antiBotEnabled = true;
345
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 342

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
341   bool processing = false;
342   mapping (address => bool) isTxLimitExempt;
343   uint256 public _maxTxAmount = 4500 * (10 ** _decimals);
344   bool public antiBotEnabled = true;
345   uint256 public cooldownTime = 30 seconds;
346
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 236

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
235   address[] memory path = new address[](2);
236   path[0] = address(token);
237   path[1] = BUSD;
238   token.approve(address(router), amount);
239   router.swapExactTokensForTokensSupportingFeeOnTransferTokens(
240
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 237

low SEVERITY

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

Source File

- Strelka.sol

Locations

```
236   path[0] = address(token);
237   path[1] = BUSD;
238   token.approve(address(router), amount);
239   router.swapExactTokensForTokensSupportingFeeOnTransferTokens(
240   amount,
241
```

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.