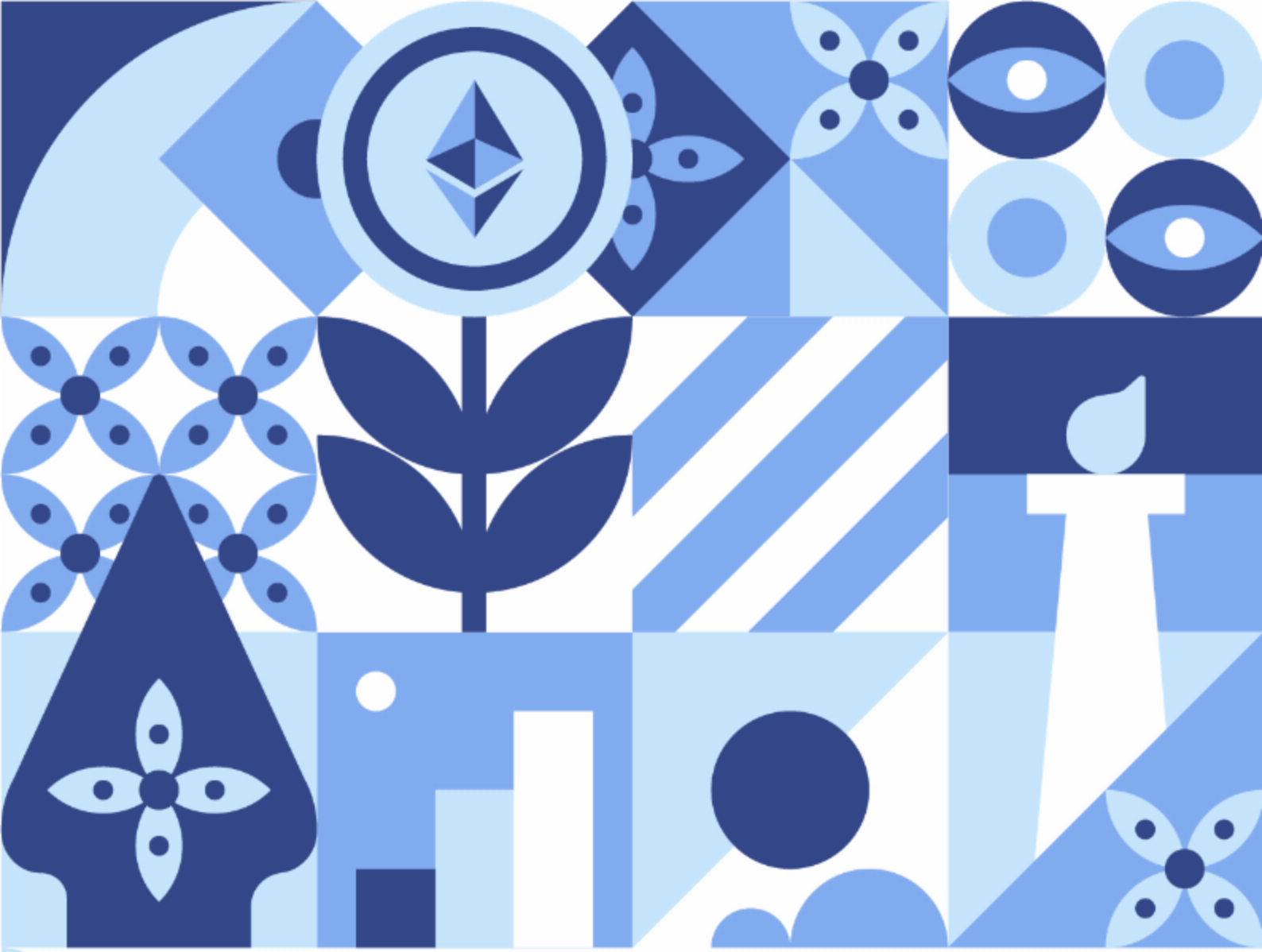

KoaCombat

Smart Contract Audit Report

SYSFIXED

19 Feb 2022

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
KoaCombat	KoaCombat	Ethereum

Addresses

Contract address	0x6769D86f9C430f5AC6d9c861a0173613F1C5544C
Contract deployer address	0xD86f63fCA24adb614f25Faf6D88f9202b4AC7c54

Project Website

https://koacombat.com/

Codebase

https://etherscan.io/address/0x6769D86f9C430f5AC6d9c861a0173613F1C5544C#code

SUMMARY

KOA COMBAT LLC leads the crypto industry with its best-in-class tokenomics, renowned cryptologists development team, 60+ years combined professional management team, top co-sponsorships, state of the art fighter NFTs, first of its kind P2E gaming platform, LIVE PPV event, staking, betting, streaming, extensive charitable giving and much more.

Contract Summary

Documentation Quality

KoaCombat 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 KoaCombat 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 119, 119, 120, 120, 123, 123, 124, 124, 244, 250, 257, 280, 294, 296, 330, 331, 335, 336, 337, 341, 342, 343, 347, 348, 349, 353, 354, 355, 371, 371, 372, 372, 373, 373, 374, 374, 375, 375, 376, 376, 376, 376, 376, 381, 387, 388, 389, 390, 391, 392, 392, 392, 392, 392, 398, 404, 406, 407, 409, 433, 438, 459, 462, 465, 466, 492, 493, 497, 564, 564, 568, 568, 577, 586, 603, 605, 606, 607 and 296.
- SWC-103 | Pragma statements can be allowed to float when a contract is intended on lines 6.
- 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 295, 296, 296, 405, 405, 406, 407, 524, 525, 587, 604, 605 and 608.

CONCLUSION

We have audited the KoaCombat project released on February-2022 to discover issues and identify potential security vulnerabilities in KoaCombat 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 KoaCombat 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 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.	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.	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	Friday Feb 18 2022 14:13:57 GMT+0000 (Coordinated Universal Time)
Finished	Saturday Feb 19 2022 10:18:36 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	KoaCombat.sol

Detected Issues

ID	Title	Severity	Status
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "%" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged

SWC-101	ARITHMETIC OPERATION "-=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged

SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged

SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged

SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-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-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-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 119

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
118
119   uint256 private _tTotal = 5e16 * 10**_decimals;
120   uint256 private _rTotal = (MAX - (MAX % _tTotal));
121
122
123
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 119

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
118
119   uint256 private _tTotal = 5e16 * 10**_decimals;
120   uint256 private _rTotal = (MAX - (MAX % _tTotal));
121
122
123
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 120

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
119   uint256 private _tTotal = 5e16 * 10**_decimals;
120   uint256 private _rTotal = (MAX - (MAX % _tTotal));
121
122
123   uint256 public swapTokensAtAmount = 500_000_000_000 * 10**_decimals;
124
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 120

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
119   uint256 private _tTotal = 5e16 * 10**_decimals;
120   uint256 private _rTotal = (MAX - (MAX % _tTotal));
121
122
123   uint256 public swapTokensAtAmount = 500_000_000_000 * 10**_decimals;
124
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 123

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
122
123   uint256 public swapTokensAtAmount = 500_000_000_000 * 10**_decimals;
124   uint256 public maxTxAmount = 5_000_000_000_000 * 10**_decimals;
125
126   // Anti Dump //
127
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 123

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
122
123   uint256 public swapTokensAtAmount = 500_000_000_000 * 10**_decimals;
124   uint256 public maxTxAmount = 5_000_000_000_000 * 10**_decimals;
125
126   // Anti Dump //
127
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 124

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
123   uint256 public swapTokensAtAmount = 500_000_000_000 * 10**_decimals;
124   uint256 public maxTxAmount = 5_000_000_000_000 * 10**_decimals;
125
126   // Anti Dump //
127   mapping (address => uint256) public _lastTrade;
128
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 124

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
123   uint256 public swapTokensAtAmount = 500_000_000_000 * 10**_decimals;
124   uint256 public maxTxAmount = 5_000_000_000_000 * 10**_decimals;
125
126   // Anti Dump //
127   mapping (address => uint256) public _lastTrade;
128
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 244

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
243   require(currentAllowance >= amount, "ERC20: transfer amount exceeds allowance");
244   _approve(sender, _msgSender(), currentAllowance - amount);
245
246   return true;
247   }
248
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 250

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
249   function increaseAllowance(address spender, uint256 addedValue) public virtual
returns (bool) {
250   _approve(_msgSender(), spender, _allowances[_msgSender()][spender] + addedValue);
251   return true;
252   }
253
254
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 257

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
256   require(currentAllowance >= subtractedValue, "ERC20: decreased allowance below
zero");
257   _approve(_msgSender(), spender, currentAllowance - subtractedValue);
258
259   return true;
260   }
261
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 280

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
279   uint256 currentRate =  _getRate();
280   return rAmount/currentRate;
281   }
282
283   function excludeFromReward(address account) public onlyOwner() {
284
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 294

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
293   require(_isExcluded[account], "Account is not excluded");
294   for (uint256 i = 0; i < _excluded.length; i++) {
295   if (_excluded[i] == account) {
296   _excluded[i] = _excluded[_excluded.length - 1];
297   _tOwned[account] = 0;
298
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 296

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
295   if (_excluded[i] == account) {
296   _excluded[i] = _excluded[_excluded.length - 1];
297   _tOwned[account] = 0;
298   _isExcluded[account] = false;
299   _excluded.pop();
300
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 330

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
329   function _reflectRfi(uint256 rRfi, uint256 tRfi) private {
330   _rTotal -=rRfi;
331   totFeesPaid.rfi +=tRfi;
332   }
333
334
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 331

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
330   _rTotal -=rRfi;
331   totFeesPaid.rfi +=tRfi;
332   }
333
334   function _takeLiquidity(uint256 rLiquidity, uint256 tLiquidity) private {
335
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 335

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
334   function _takeLiquidity(uint256 rLiquidity, uint256 tLiquidity) private {
335   totFeesPaid.liquidity +=tLiquidity;
336   if(_isExcluded[address(this)]) _tOwned[address(this)]+=tLiquidity;
337   _rOwned[address(this)] +=rLiquidity;
338   }
339
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 336

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
335   totFeesPaid.liquidity +=tLiquidity;
336   if(_isExcluded[address(this)]) _tOwned[address(this)]+=tLiquidity;
337   _rOwned[address(this)] +=rLiquidity;
338   }
339
340
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 337

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
336   if(_isExcluded[address(this)]) _tOwned[address(this)]+=tLiquidity;
337   _rOwned[address(this)] +=rLiquidity;
338   }
339
340   function _takeTreasury(uint256 rTreasury, uint256 tTreasury) private {
341
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 341

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
340   function _takeTreasury(uint256 rTreasury, uint256 tTreasury) private {
341   totFeesPaid.treasury +=tTreasury;
342   if(_isExcluded[treasuryAddress]) _tOwned[treasuryAddress]+=tTreasury;
343   _rOwned[treasuryAddress] +=rTreasury;
344   }
345
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 342

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
341   totFeesPaid.treasury +=tTreasury;
342   if(_isExcluded[treasuryAddress]) _tOwned[treasuryAddress]+=tTreasury;
343   _rOwned[treasuryAddress] +=rTreasury;
344   }
345
346
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 343

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
342   if(_isExcluded[treasuryAddress]) _tOwned[treasuryAddress]+=tTreasury;
343   _rOwned[treasuryAddress] +=rTreasury;
344   }
345
346   function _takeCharity(uint256 rCharity, uint256 tCharity) private{
347
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 347

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
346   function _takeCharity(uint256 rCharity, uint256 tCharity) private{
347   totFeesPaid.charity +=tCharity;
348   if(_isExcluded[charityAddress]) _tOwned[charityAddress]+=tCharity;
349   _rOwned[charityAddress] +=rCharity;
350   }
351
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 348

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
347   totFeesPaid.charity +=tCharity;
348   if(_isExcluded[charityAddress]) _tOwned[charityAddress]+=tCharity;
349   _rOwned[charityAddress] +=rCharity;
350   }
351
352
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 349

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
348   if(_isExcluded[charityAddress]) _tOwned[charityAddress]+=tCharity;
349   _rOwned[charityAddress] +=rCharity;
350   }
351
352   function _takeBurn(uint256 rBurn, uint256 tBurn) private{
353
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 353

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
352   function _takeBurn(uint256 rBurn, uint256 tBurn) private{
353   totFeesPaid.burn +=tBurn;
354   if(_isExcluded[charityAddress])_tOwned[burnAddress]+=tBurn;
355   _rOwned[burnAddress] +=rBurn;
356   }
357
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 354

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
353   totFeesPaid.burn +=tBurn;
354   if(_isExcluded[charityAddress])_tOwned[burnAddress]+=tBurn;
355   _rOwned[burnAddress] +=rBurn;
356   }
357
358
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 355

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
354   if(_isExcluded[charityAddress])_tOwned[burnAddress]+=tBurn;
355   _rOwned[burnAddress] +=rBurn;
356   }
357
358   function _getValues(uint256 tAmount, bool takeFee) private view returns
(valuesFromGetValues memory to_return) {
359
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 371

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
370
371   s.tRfi = tAmount*taxes.rfi/1000;
372   s.tTreasury = tAmount*taxes.treasury/1000;
373   s.tCharity = tAmount*taxes.charity/1000;
374   s.tBurn = tAmount*taxes.burn/1000;
375
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 371

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
370
371   s.tRfi = tAmount*taxes.rfi/1000;
372   s.tTreasury = tAmount*taxes.treasury/1000;
373   s.tCharity = tAmount*taxes.charity/1000;
374   s.tBurn = tAmount*taxes.burn/1000;
375
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 372

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
371   s.tRfi = tAmount*taxes.rfi/1000;
372   s.tTreasury = tAmount*taxes.treasury/1000;
373   s.tCharity = tAmount*taxes.charity/1000;
374   s.tBurn = tAmount*taxes.burn/1000;
375   s.tLiquidity = tAmount*taxes.liquidity/1000;
376
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 372

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
371   s.tRfi = tAmount*taxes.rfi/1000;
372   s.tTreasury = tAmount*taxes.treasury/1000;
373   s.tCharity = tAmount*taxes.charity/1000;
374   s.tBurn = tAmount*taxes.burn/1000;
375   s.tLiquidity = tAmount*taxes.liquidity/1000;
376
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 373

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
372   s.tTreasury = tAmount*taxes.treasury/1000;
373   s.tCharity = tAmount*taxes.charity/1000;
374   s.tBurn = tAmount*taxes.burn/1000;
375   s.tLiquidity = tAmount*taxes.liquidity/1000;
376   s.tTransferAmount = tAmount-s.tRfi-s.tTreasury-s.tLiquidity-s.tCharity-s.tBurn;
377
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 373

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
372   s.tTreasury = tAmount*taxes.treasury/1000;
373   s.tCharity = tAmount*taxes.charity/1000;
374   s.tBurn = tAmount*taxes.burn/1000;
375   s.tLiquidity = tAmount*taxes.liquidity/1000;
376   s.tTransferAmount = tAmount-s.tRfi-s.tTreasury-s.tLiquidity-s.tCharity-s.tBurn;
377
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 374

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
373   s.tCharity = tAmount*taxes.charity/1000;
374   s.tBurn = tAmount*taxes.burn/1000;
375   s.tLiquidity = tAmount*taxes.liquidity/1000;
376   s.tTransferAmount = tAmount-s.tRfi-s.tTreasury-s.tLiquidity-s.tCharity-s.tBurn;
377   return s;
378
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 374

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
373   s.tCharity = tAmount*taxes.charity/1000;
374   s.tBurn = tAmount*taxes.burn/1000;
375   s.tLiquidity = tAmount*taxes.liquidity/1000;
376   s.tTransferAmount = tAmount-s.tRfi-s.tTreasury-s.tLiquidity-s.tCharity-s.tBurn;
377   return s;
378
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 375

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
374   s.tBurn = tAmount*taxes.burn/1000;
375   s.tLiquidity = tAmount*taxes.liquidity/1000;
376   s.tTransferAmount = tAmount-s.tRfi-s.tTreasury-s.tLiquidity-s.tCharity-s.tBurn;
377   return s;
378   }
379
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 375

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
374   s.tBurn = tAmount*taxes.burn/1000;
375   s.tLiquidity = tAmount*taxes.liquidity/1000;
376   s.tTransferAmount = tAmount-s.tRfi-s.tTreasury-s.tLiquidity-s.tCharity-s.tBurn;
377   return s;
378   }
379
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 376

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
375   s.tLiquidity = tAmount*taxes.liquidity/1000;
376   s.tTransferAmount = tAmount-s.tRfi-s.tTreasury-s.tLiquidity-s.tCharity-s.tBurn;
377   return s;
378   }
379
380
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 376

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
375   s.tLiquidity = tAmount*taxes.liquidity/1000;
376   s.tTransferAmount = tAmount-s.tRfi-s.tTreasury-s.tLiquidity-s.tCharity-s.tBurn;
377   return s;
378   }
379
380
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 376

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
375   s.tLiquidity = tAmount*taxes.liquidity/1000;
376   s.tTransferAmount = tAmount-s.tRfi-s.tTreasury-s.tLiquidity-s.tCharity-s.tBurn;
377   return s;
378   }
379
380
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 376

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
375   s.tLiquidity = tAmount*taxes.liquidity/1000;
376   s.tTransferAmount = tAmount-s.tRfi-s.tTreasury-s.tLiquidity-s.tCharity-s.tBurn;
377   return s;
378   }
379
380
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 376

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
375   s.tLiquidity = tAmount*taxes.liquidity/1000;
376   s.tTransferAmount = tAmount-s.tRfi-s.tTreasury-s.tLiquidity-s.tCharity-s.tBurn;
377   return s;
378   }
379
380
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 381

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
380   function _getRValues(valuesFromGetValues memory s, uint256 tAmount, bool takeFee,
uint256 currentRate) private pure returns (uint256 rAmount, uint256 rTransferAmount,
uint256 rRfi,uint256 rTreasury,uint256 rCharity,uint256 rBurn,uint256 rLiquidity) {
381   rAmount = tAmount*currentRate;
382
383   if(!takeFee) {
384   return(rAmount, rAmount, 0,0,0,0,0);
385
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 387

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
386
387   rRfi = s.tRfi*currentRate;
388   rTreasury = s.tTreasury*currentRate;
389   rLiquidity = s.tLiquidity*currentRate;
390   rCharity = s.tCharity*currentRate;
391
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 388

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
387   rRfi = s.tRfi*currentRate;
388   rTreasury = s.tTreasury*currentRate;
389   rLiquidity = s.tLiquidity*currentRate;
390   rCharity = s.tCharity*currentRate;
391   rBurn = s.tBurn*currentRate;
392
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 389

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
388   rTreasury = s.tTreasury*currentRate;
389   rLiquidity = s.tLiquidity*currentRate;
390   rCharity = s.tCharity*currentRate;
391   rBurn = s.tBurn*currentRate;
392   rTransferAmount =  rAmount-rRfi-rTreasury-rLiquidity-rCharity-rBurn;
393
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 390

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
389   rLiquidity = s.tLiquidity*currentRate;
390   rCharity = s.tCharity*currentRate;
391   rBurn = s.tBurn*currentRate;
392   rTransferAmount =  rAmount-rRfi-rTreasury-rLiquidity-rCharity-rBurn;
393   return (rAmount, rTransferAmount, rRfi,rTreasury,rCharity,rBurn,rLiquidity);
394
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 391

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
390   rCharity = s.tCharity*currentRate;
391   rBurn = s.tBurn*currentRate;
392   rTransferAmount =  rAmount-rRfi-rTreasury-rLiquidity-rCharity-rBurn;
393   return (rAmount, rTransferAmount, rRfi,rTreasury,rCharity,rBurn,rLiquidity);
394   }
395
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 392

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
391   rBurn = s.tBurn*currentRate;
392   rTransferAmount =  rAmount-rRfi-rTreasury-rLiquidity-rCharity-rBurn;
393   return (rAmount, rTransferAmount, rRfi,rTreasury,rCharity,rBurn,rLiquidity);
394   }
395
396
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 392

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
391   rBurn = s.tBurn*currentRate;
392   rTransferAmount =  rAmount-rRfi-rTreasury-rLiquidity-rCharity-rBurn;
393   return (rAmount, rTransferAmount, rRfi,rTreasury,rCharity,rBurn,rLiquidity);
394   }
395
396
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 392

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
391   rBurn = s.tBurn*currentRate;
392   rTransferAmount =  rAmount-rRfi-rTreasury-rLiquidity-rCharity-rBurn;
393   return (rAmount, rTransferAmount, rRfi,rTreasury,rCharity,rBurn,rLiquidity);
394   }
395
396
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 392

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
391   rBurn = s.tBurn*currentRate;
392   rTransferAmount =  rAmount-rRfi-rTreasury-rLiquidity-rCharity-rBurn;
393   return (rAmount, rTransferAmount, rRfi,rTreasury,rCharity,rBurn,rLiquidity);
394   }
395
396
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 392

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
391   rBurn = s.tBurn*currentRate;
392   rTransferAmount =  rAmount-rRfi-rTreasury-rLiquidity-rCharity-rBurn;
393   return (rAmount, rTransferAmount, rRfi,rTreasury,rCharity,rBurn,rLiquidity);
394   }
395
396
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 398

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
397   (uint256 rSupply, uint256 tSupply) = _getCurrentSupply();
398   return rSupply/tSupply;
399   }
400
401   function _getCurrentSupply() private view returns(uint256, uint256) {
402
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 404

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
403   uint256 tSupply = _tTotal;
404   for (uint256 i = 0; i < _excluded.length; i++) {
405   if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return
(_rTotal, _tTotal);
406   rSupply = rSupply-_rOwned[_excluded[i]];
407   tSupply = tSupply-_tOwned[_excluded[i]];
408
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 406

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
405   if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return
(_rTotal, _tTotal);
406   rSupply = rSupply-_rOwned[_excluded[i]];
407   tSupply = tSupply-_tOwned[_excluded[i]];
408   }
409   if (rSupply < _rTotal/_tTotal) return (_rTotal, _tTotal);
410
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 407

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
406   rSupply = rSupply-_rOwned[_excluded[i]];
407   tSupply = tSupply-_tOwned[_excluded[i]];
408   }
409   if (rSupply < _rTotal/_tTotal) return (_rTotal, _tTotal);
410   return (rSupply, tSupply);
411
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 409

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
408   }
409   if (rSupply < _rTotal/_tTotal) return (_rTotal, _tTotal);
410   return (rSupply, tSupply);
411   }
412
413
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 433

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
432   if(from != pair && coolDownEnabled){
433   uint256 timePassed = block.timestamp - _lastTrade[from];
434   require(timePassed > coolDownTime, "You must wait coolDownTime");
435   _lastTrade[from] = block.timestamp;
436   }
437
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 438

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
437   if(to != pair && coolDownEnabled){
438   uint256 timePassed2 = block.timestamp - _lastTrade[to];
439   require(timePassed2 > coolDownTime, "You must wait coolDownTime");
440   _lastTrade[to] = block.timestamp;
441   }
442
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 459

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
458   if (_isExcluded[sender] ) {  //from excluded
459   _tOwned[sender] = _tOwned[sender]-tAmount;
460   }
461   if (_isExcluded[recipient]) { //to excluded
462   _tOwned[recipient] = _tOwned[recipient]+s.tTransferAmount;
463
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 462

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
461   if (_isExcluded[recipient]) { //to excluded
462   _tOwned[recipient] = _tOwned[recipient]+s.tTransferAmount;
463   }
464
465   _rOwned[sender] = _rOwned[sender]-s.rAmount;
466
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 465

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
464
465   _rOwned[sender] = _rOwned[sender]-s.rAmount;
466   _rOwned[recipient] = _rOwned[recipient]+s.rTransferAmount;
467
468   if(s.rRfi > 0 || s.tRfi > 0) _reflectRfi(s.rRfi, s.tRfi);
469
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 466

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
465   _rOwned[sender] = _rOwned[sender]-s.rAmount;
466   _rOwned[recipient] = _rOwned[recipient]+s.rTransferAmount;
467
468   if(s.rRfi > 0 || s.tRfi > 0) _reflectRfi(s.rRfi, s.tRfi);
469   if(s.rLiquidity > 0 || s.tLiquidity > 0) {
470
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 492

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
491   // Split the contract balance into halves
492   uint256 tokensToAddLiquidityWith = tokens / 2;
493   uint256 toSwap = tokens - tokensToAddLiquidityWith;
494
495   uint256 initialBalance = address(this).balance;
496
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 493

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
492   uint256 tokensToAddLiquidityWith = tokens / 2;
493   uint256 toSwap = tokens - tokensToAddLiquidityWith;
494
495   uint256 initialBalance = address(this).balance;
496   swapTokensForETH(toSwap);
497
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 497

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
496   swapTokensForETH(toSwap);
497   uint256 ETHToAddLiquidityWith = address(this).balance - initialBalance;
498
499   if(ETHToAddLiquidityWith > 0){
500   // Add liquidity to pancake
501
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 564

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
563   function updatMaxTxAmt(uint256 amount) external onlyOwner{
564   maxTxAmount = amount * 10**_decimals;
565   }
566
567   function updateSwapTokensAtAmount(uint256 amount) external onlyOwner{
568
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 564

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
563   function updatMaxTxAmt(uint256 amount) external onlyOwner{
564   maxTxAmount = amount * 10**_decimals;
565   }
566
567   function updateSwapTokensAtAmount(uint256 amount) external onlyOwner{
568
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 568

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
567   function updateSwapTokensAtAmount(uint256 amount) external onlyOwner{
568   swapTokensAtAmount = amount * 10**_decimals;
569   }
570
571   function updateSwapEnabled(bool _enabled) external onlyOwner{
572
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 568

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
567   function updateSwapTokensAtAmount(uint256 amount) external onlyOwner{
568   swapTokensAtAmount = amount * 10**_decimals;
569   }
570
571   function updateSwapEnabled(bool _enabled) external onlyOwner{
572
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 577

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
576   coolDownEnabled = _enabled;
577   coolDownTime = _timeInSeconds * 1 seconds;
578   }
579
580   function setAntibot(address account, bool state) external onlyOwner{
581
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 586

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
585   function bulkAntiBot(address[] memory accounts, bool state) external onlyOwner{
586   for(uint256 i = 0; i < accounts.length; i++){
587   _isBot[accounts[i]] = state;
588   }
589   }
590
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 603

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
602   address sender = msg.sender;
603   for(uint256 i; i<recipients.length; i++){
604   address recipient = recipients[i];
605   uint256 rAmount = amounts[i]*_getRate();
606   _rOwned[sender] = _rOwned[sender]- rAmount;
607
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 605

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
604   address recipient = recipients[i];
605   uint256 rAmount = amounts[i]*_getRate();
606   _rOwned[sender] = _rOwned[sender]- rAmount;
607   _rOwned[recipient] = _rOwned[recipient] + rAmount;
608   emit Transfer(sender, recipient, amounts[i]);
609
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 606

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
605   uint256 rAmount = amounts[i]*_getRate();
606   _rOwned[sender] = _rOwned[sender]- rAmount;
607   _rOwned[recipient] = _rOwned[recipient] + rAmount;
608   emit Transfer(sender, recipient, amounts[i]);
609   }
610
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 607

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
606   _rOwned[sender] = _rOwned[sender]- rAmount;
607   _rOwned[recipient] = _rOwned[recipient] + rAmount;
608   emit Transfer(sender, recipient, amounts[i]);
609   }
610   }
611
```

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

LINE 296

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
295   if (_excluded[i] == account) {
296   _excluded[i] = _excluded[_excluded.length - 1];
297   _tOwned[account] = 0;
298   _isExcluded[account] = false;
299   _excluded.pop();
300
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 6

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

- KoaCombat.sol

Locations

```
5   // SPDX-License-Identifier: NOLICENSE
6   pragma solidity ^0.8.10;
7
8   interface IERC20 {
9   function totalSupply() external view returns (uint256);
10
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 295

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
294   for (uint256 i = 0; i < _excluded.length; i++) {
295   if (_excluded[i] == account) {
296   _excluded[i] = _excluded[_excluded.length - 1];
297   _tOwned[account] = 0;
298   _isExcluded[account] = false;
299
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 296

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
295   if (_excluded[i] == account) {
296   _excluded[i] = _excluded[_excluded.length - 1];
297   _tOwned[account] = 0;
298   _isExcluded[account] = false;
299   _excluded.pop();
300
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 296

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
295   if (_excluded[i] == account) {
296   _excluded[i] = _excluded[_excluded.length - 1];
297   _tOwned[account] = 0;
298   _isExcluded[account] = false;
299   _excluded.pop();
300
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 405

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
404   for (uint256 i = 0; i < _excluded.length; i++) {
405   if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return
(_rTotal, _tTotal);
406   rSupply = rSupply-_rOwned[_excluded[i]];
407   tSupply = tSupply-_tOwned[_excluded[i]];
408   }
409
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 405

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
404   for (uint256 i = 0; i < _excluded.length; i++) {
405   if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return
(_rTotal, _tTotal);
406   rSupply = rSupply-_rOwned[_excluded[i]];
407   tSupply = tSupply-_tOwned[_excluded[i]];
408   }
409
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 406

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
405   if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return
(_rTotal, _tTotal);
406   rSupply = rSupply-_rOwned[_excluded[i]];
407   tSupply = tSupply-_tOwned[_excluded[i]];
408   }
409   if (rSupply < _rTotal/_tTotal) return (_rTotal, _tTotal);
410
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 407

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
406   rSupply = rSupply-_rOwned[_excluded[i]];
407   tSupply = tSupply-_tOwned[_excluded[i]];
408   }
409   if (rSupply < _rTotal/_tTotal) return (_rTotal, _tTotal);
410   return (rSupply, tSupply);
411
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 524

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
523   address[] memory path = new address[](2);
524   path[0] = address(this);
525   path[1] = router.WETH();
526
527   _approve(address(this), address(router), tokenAmount);
528
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 525

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
524   path[0] = address(this);
525   path[1] = router.WETH();
526
527   _approve(address(this), address(router), tokenAmount);
528
529
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 587

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
586   for(uint256 i = 0; i < accounts.length; i++){
587   _isBot[accounts[i]] = state;
588   }
589   }
590
591
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 604

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
603   for(uint256 i; i<recipients.length; i++){
604   address recipient = recipients[i];
605   uint256 rAmount = amounts[i]*_getRate();
606   _rOwned[sender] = _rOwned[sender]- rAmount;
607   _rOwned[recipient] = _rOwned[recipient] + rAmount;
608
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 605

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
604   address recipient = recipients[i];
605   uint256 rAmount = amounts[i]*_getRate();
606   _rOwned[sender] = _rOwned[sender]- rAmount;
607   _rOwned[recipient] = _rOwned[recipient] + rAmount;
608   emit Transfer(sender, recipient, amounts[i]);
609
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 608

low SEVERITY

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

Source File

- KoaCombat.sol

Locations

```
607   _rOwned[recipient] = _rOwned[recipient] + rAmount;
608   emit Transfer(sender, recipient, amounts[i]);
609   }
610   }
611
612
```

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.