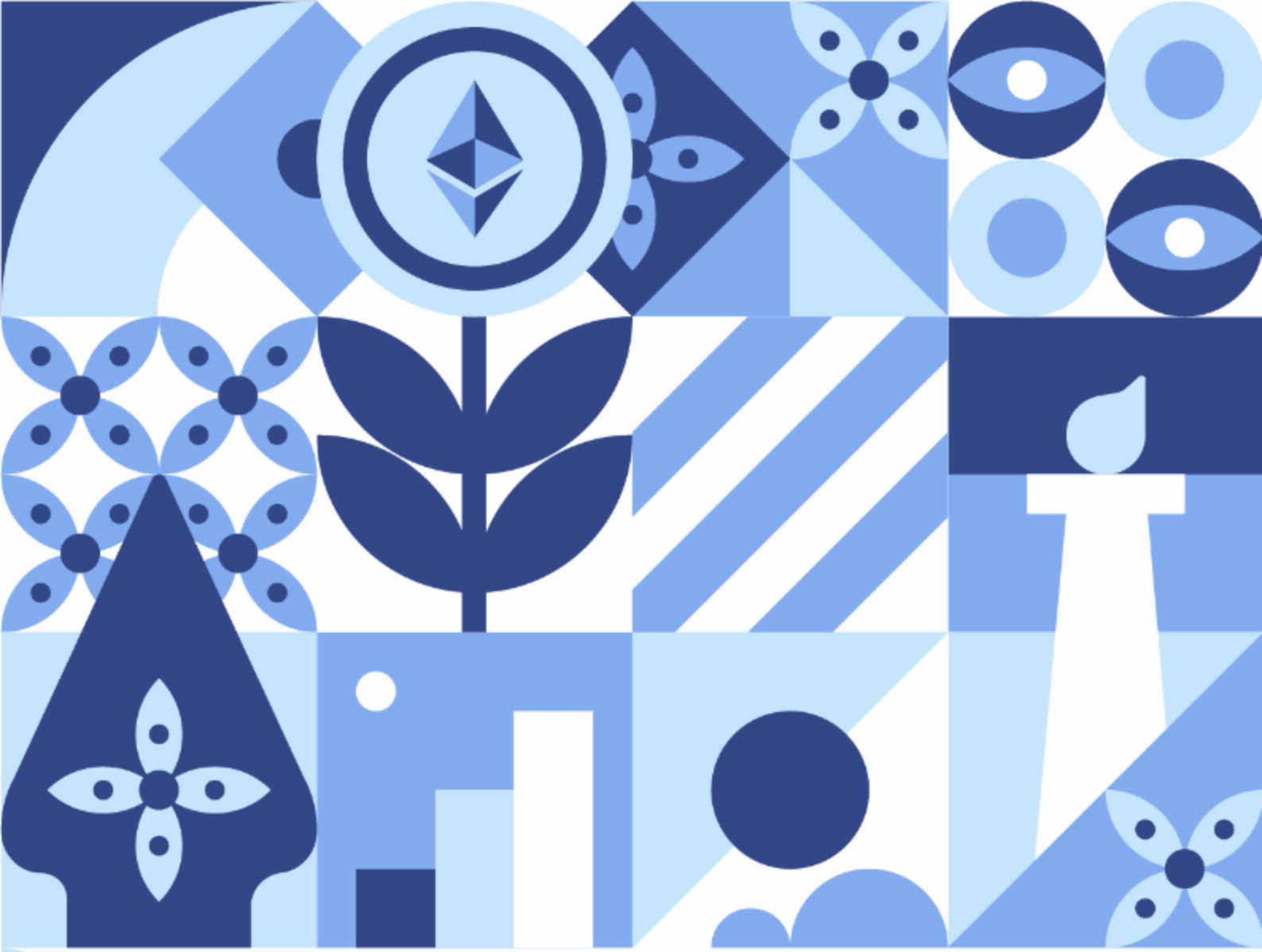

AnimeVerse

Smart Contract Audit Report

SYSFIXED

25 May 2022

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
AnimeVerse	Anime	Ethereum

Addresses

Contract address	0x5a8F92addfe1Cd48B51E1FA926144C0918DBAb67
Contract deployer address	0x8DA699d90a052B62EF97f79E16f4C75324723903

Project Website

https://www.animeversetoken.com/

Codebase

https://etherscan.io/address/0x5a8F92addfe1Cd48B51E1FA926144C0918DBAb67#code

SUMMARY

As bad actors entered the DeFi space, the market started to head into a downward trend leaving many investors in a vulnerable position. AnimeVerse has been assembled to defend the DeFi space from corruption and market manipulation we often see. Order needs to be restored and AnimeVerse is here to do it.

Contract Summary

Documentation Quality

AnimeVerse 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 AnimeVerse 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 126, 127, 129, 193, 194, 196 and 207.
- SWC-101 | It is recommended to use vetted safe math libraries for arithmetic operations consistently on lines 141, 141, 166, 166, 189, 189, 190, 190, 191, 191, 333, 358, 418, 449, 449, 450, 450, 458, 468, 468, 469, 478, 478, 478, 479, 479, 479, 480, 480, 481, 481, 485, 485, 485, 486, 486, 490, 490, 494, 494, 498, 498, 502, 502, 503, 503, 520, 520, 570, 582, 582, 618, 623, 657, 657, 659, 662, 675, 675, 675, 676, 691, 691, 705, 706, 708, 708, 709, 738, 740 and 740.
- 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 419, 679, 680, 739, 740 and 740.
- SWC-115 | tx.origin should not be used for authorization, use msg.sender instead on lines 532.
- SWC-120 | It is recommended to use external sources of randomness via oracles on lines 446.

CONCLUSION

We have audited the AnimeVerse project released on May 2022 to discover issues and identify potential security vulnerabilities in AnimeVerse 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 AnimeVerse smart contract code do not pose a considerable risk. The writing of the contract is close to the standard of writing contracts in general. The low-risk issues found are some arithmetic operation issues, a floating pragma is set, a state variable visibility is not set, weak sources of randomness, tx.origin as a part of authorization control and out of bounds array access which the index access expression can cause an exception in case of the use of an invalid array index value.

AUDIT RESULT

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

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

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

SMART CONTRACT ANALYSIS

Started	Tuesday May 24 2022 08:44:54 GMT+0000 (Coordinated Universal Time)
Finished	Wednesday May 25 2022 20:00:49 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	AnimeVerse.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

SYSFIXED

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

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

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

SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-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-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-115	USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-120	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	low	acknowledged

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 141

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
140
141   uint256 constant private _tTotal = startingSupply * 10**_decimals;
142
143   struct Fees {
144   uint16 buyFee;
145
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 141

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
140
141   uint256 constant private _tTotal = startingSupply * 10**_decimals;
142
143   struct Fees {
144   uint16 buyFee;
145
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 166

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
165   marketing: 2580,
166   total: 800 + 600 + 2580
167   });
168
169   uint256 constant public maxBuyTaxes = 2000;
170
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 166

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
165   marketing: 2580,
166   total: 800 + 600 + 2580
167   });
168
169   uint256 constant public maxBuyTaxes = 2000;
170
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 189

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
188
189   uint256 private _maxTxAmountBuy = (_tTotal * 15) / 1000;
190   uint256 private _maxTxAmountSell = (_tTotal * 75) / 10000;
191   uint256 private _maxWalletSize = (_tTotal * 15) / 1000;
192
193
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 189

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
188
189   uint256 private _maxTxAmountBuy = (_tTotal * 15) / 1000;
190   uint256 private _maxTxAmountSell = (_tTotal * 75) / 10000;
191   uint256 private _maxWalletSize = (_tTotal * 15) / 1000;
192
193
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 190

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
189   uint256 private _maxTxAmountBuy = (_tTotal * 15) / 1000;
190   uint256 private _maxTxAmountSell = (_tTotal * 75) / 10000;
191   uint256 private _maxWalletSize = (_tTotal * 15) / 1000;
192
193   Cashier reflector;
194
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 190

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
189   uint256 private _maxTxAmountBuy = (_tTotal * 15) / 1000;
190   uint256 private _maxTxAmountSell = (_tTotal * 75) / 10000;
191   uint256 private _maxWalletSize = (_tTotal * 15) / 1000;
192
193   Cashier reflector;
194
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 191

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
190   uint256 private _maxTxAmountSell = (_tTotal * 75) / 10000;
191   uint256 private _maxWalletSize = (_tTotal * 15) / 1000;
192
193   Cashier reflector;
194   uint256 reflectorGas = 300000;
195
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 191

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
190   uint256 private _maxTxAmountSell = (_tTotal * 75) / 10000;
191   uint256 private _maxWalletSize = (_tTotal * 15) / 1000;
192
193   Cashier reflector;
194   uint256 reflectorGas = 300000;
195
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 333

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
332   if (_allowances[sender][msg.sender] != type(uint256).max) {
333   _allowances[sender][msg.sender] -= amount;
334   }
335
336   return _transfer(sender, recipient, amount);
337
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 358

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
357   if (timeSinceLastPair != 0) {
358   require(block.timestamp - timeSinceLastPair > 3 days, "Cannot set a new pair this
week!");
359   }
360   lpPairs[pair] = true;
361   timeSinceLastPair = block.timestamp;
362
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 418

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
417   antiSnipe.setBlacklistEnabledMultiple(accounts, enabled);
418   for(uint256 i = 0; i < accounts.length; i++){
419   setDividendExcluded(accounts[i], enabled);
420   }
421   }
422
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 449

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
448   tradingEnabled = true;
449   swapThreshold = (balanceOf(lpPair) * 10) / 10000;
450   swapAmount = (balanceOf(lpPair) * 25) / 10000;
451   }
452
453
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 449

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
448   tradingEnabled = true;
449   swapThreshold = (balanceOf(lpPair) * 10) / 10000;
450   swapAmount = (balanceOf(lpPair) * 25) / 10000;
451   }
452
453
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 450

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
449   swapThreshold = (balanceOf(lpPair) * 10) / 10000;
450   swapAmount = (balanceOf(lpPair) * 25) / 10000;
451   }
452
453   function setTaxes(uint16 buyFee, uint16 sellFee, uint16 transferFee) external
onlyOwner {
454
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 450

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
449   swapThreshold = (balanceOf(lpPair) * 10) / 10000;
450   swapAmount = (balanceOf(lpPair) * 25) / 10000;
451   }
452
453   function setTaxes(uint16 buyFee, uint16 sellFee, uint16 transferFee) external
onlyOwner {
454
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 458

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
457   "Cannot exceed maximums.");
458   require(buyFee + sellFee <= maxRoundtripTax, "Cannot exceed roundtrip maximum.");
459   _taxRates.buyFee = buyFee;
460   _taxRates.sellFee = sellFee;
461   _taxRates.transferFee = transferFee;
462
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 468

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
467   _ratios.marketing = marketing;
468   _ratios.total = rewards + liquidity + marketing;
469   uint256 total = _taxRates.buyFee + _taxRates.sellFee;
470   require(_ratios.total <= total, "Cannot exceed sum of buy and sell fees.");
471   }
472
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 468

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
467   _ratios.marketing = marketing;
468   _ratios.total = rewards + liquidity + marketing;
469   uint256 total = _taxRates.buyFee + _taxRates.sellFee;
470   require(_ratios.total <= total, "Cannot exceed sum of buy and sell fees.");
471   }
472
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 469

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
468   _ratios.total = rewards + liquidity + marketing;
469   uint256 total = _taxRates.buyFee + _taxRates.sellFee;
470   require(_ratios.total <= total, "Cannot exceed sum of buy and sell fees.");
471   }
472
473
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 478

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
477   function setMaxTxPercents(uint256 percentBuy, uint256 divisorBuy, uint256
percentSell, uint256 divisorSell) external onlyOwner {
478   require((_tTotal * percentBuy) / divisorBuy >= (_tTotal / 1000), "Max Transaction
amt must be above 0.1% of total supply.");
479   require((_tTotal * percentSell) / divisorSell >= (_tTotal / 1000), "Max Transaction
amt must be above 0.1% of total supply.");
480   _maxTxAmountBuy = (_tTotal * percentBuy) / divisorBuy;
481   _maxTxAmountSell = (_tTotal * percentSell) / divisorSell;
482
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 478

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
477   function setMaxTxPercents(uint256 percentBuy, uint256 divisorBuy, uint256
percentSell, uint256 divisorSell) external onlyOwner {
478   require((_tTotal * percentBuy) / divisorBuy >= (_tTotal / 1000), "Max Transaction
amt must be above 0.1% of total supply.");
479   require((_tTotal * percentSell) / divisorSell >= (_tTotal / 1000), "Max Transaction
amt must be above 0.1% of total supply.");
480   _maxTxAmountBuy = (_tTotal * percentBuy) / divisorBuy;
481   _maxTxAmountSell = (_tTotal * percentSell) / divisorSell;
482
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 478

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
477   function setMaxTxPercents(uint256 percentBuy, uint256 divisorBuy, uint256
percentSell, uint256 divisorSell) external onlyOwner {
478   require((_tTotal * percentBuy) / divisorBuy >= (_tTotal / 1000), "Max Transaction
amt must be above 0.1% of total supply.");
479   require((_tTotal * percentSell) / divisorSell >= (_tTotal / 1000), "Max Transaction
amt must be above 0.1% of total supply.");
480   _maxTxAmountBuy = (_tTotal * percentBuy) / divisorBuy;
481   _maxTxAmountSell = (_tTotal * percentSell) / divisorSell;
482
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 479

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
478   require((_tTotal * percentBuy) / divisorBuy >= (_tTotal / 1000), "Max Transaction
amt must be above 0.1% of total supply.");
479   require((_tTotal * percentSell) / divisorSell >= (_tTotal / 1000), "Max Transaction
amt must be above 0.1% of total supply.");
480   _maxTxAmountBuy = (_tTotal * percentBuy) / divisorBuy;
481   _maxTxAmountSell = (_tTotal * percentSell) / divisorSell;
482   }
483
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 479

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
478   require((_tTotal * percentBuy) / divisorBuy >= (_tTotal / 1000), "Max Transaction
amt must be above 0.1% of total supply.");
479   require((_tTotal * percentSell) / divisorSell >= (_tTotal / 1000), "Max Transaction
amt must be above 0.1% of total supply.");
480   _maxTxAmountBuy = (_tTotal * percentBuy) / divisorBuy;
481   _maxTxAmountSell = (_tTotal * percentSell) / divisorSell;
482   }
483
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 479

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
478   require((_tTotal * percentBuy) / divisorBuy >= (_tTotal / 1000), "Max Transaction
amt must be above 0.1% of total supply.");
479   require((_tTotal * percentSell) / divisorSell >= (_tTotal / 1000), "Max Transaction
amt must be above 0.1% of total supply.");
480   _maxTxAmountBuy = (_tTotal * percentBuy) / divisorBuy;
481   _maxTxAmountSell = (_tTotal * percentSell) / divisorSell;
482   }
483
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 480

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
479   require((_tTotal * percentSell) / divisorSell >= (_tTotal / 1000), "Max Transaction
amt must be above 0.1% of total supply.");
480   _maxTxAmountBuy = (_tTotal * percentBuy) / divisorBuy;
481   _maxTxAmountSell = (_tTotal * percentSell) / divisorSell;
482   }
483
484
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 480

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
479   require((_tTotal * percentSell) / divisorSell >= (_tTotal / 1000), "Max Transaction
amt must be above 0.1% of total supply.");
480   _maxTxAmountBuy = (_tTotal * percentBuy) / divisorBuy;
481   _maxTxAmountSell = (_tTotal * percentSell) / divisorSell;
482   }
483
484
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 481

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
480   _maxTxAmountBuy = (_tTotal * percentBuy) / divisorBuy;
481   _maxTxAmountSell = (_tTotal * percentSell) / divisorSell;
482   }
483
484   function setMaxWalletSize(uint256 percent, uint256 divisor) external onlyOwner {
485
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 481

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
480   _maxTxAmountBuy = (_tTotal * percentBuy) / divisorBuy;
481   _maxTxAmountSell = (_tTotal * percentSell) / divisorSell;
482   }
483
484   function setMaxWalletSize(uint256 percent, uint256 divisor) external onlyOwner {
485
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 485

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
484   function setMaxWalletSize(uint256 percent, uint256 divisor) external onlyOwner {
485   require((_tTotal * percent) / divisor >= (_tTotal / 1000), "Max Wallet amt must be
above 0.1% of total supply.");
486   _maxWalletSize = (_tTotal * percent) / divisor;
487   }
488
489
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 485

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
484   function setMaxWalletSize(uint256 percent, uint256 divisor) external onlyOwner {
485   require((_tTotal * percent) / divisor >= (_tTotal / 1000), "Max Wallet amt must be
above 0.1% of total supply.");
486   _maxWalletSize = (_tTotal * percent) / divisor;
487   }
488
489
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 485

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
484   function setMaxWalletSize(uint256 percent, uint256 divisor) external onlyOwner {
485   require((_tTotal * percent) / divisor >= (_tTotal / 1000), "Max Wallet amt must be
above 0.1% of total supply.");
486   _maxWalletSize = (_tTotal * percent) / divisor;
487   }
488
489
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 486

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
485   require((_tTotal * percent) / divisor >= (_tTotal / 1000), "Max Wallet amt must be
above 0.1% of total supply.");
486   _maxWalletSize = (_tTotal * percent) / divisor;
487   }
488
489   function getMaxTXBuy() public view returns (uint256) {
490
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 486

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
485   require((_tTotal * percent) / divisor >= (_tTotal / 1000), "Max Wallet amt must be
above 0.1% of total supply.");
486   _maxWalletSize = (_tTotal * percent) / divisor;
487   }
488
489   function getMaxTXBuy() public view returns (uint256) {
490
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 490

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
489   function getMaxTXBuy() public view returns (uint256) {
490   return _maxTxAmountBuy / (10**_decimals);
491   }
492
493   function getMaxTXSell() public view returns (uint256) {
494
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 490

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
489   function getMaxTXBuy() public view returns (uint256) {
490   return _maxTxAmountBuy / (10**_decimals);
491   }
492
493   function getMaxTXSell() public view returns (uint256) {
494
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 494

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
493   function getMaxTXSell() public view returns (uint256) {
494   return _maxTxAmountSell / (10**_decimals);
495   }
496
497   function getMaxWallet() public view returns (uint256) {
498
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 494

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
493   function getMaxTXSell() public view returns (uint256) {
494   return _maxTxAmountSell / (10**_decimals);
495   }
496
497   function getMaxWallet() public view returns (uint256) {
498
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 498

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
497   function getMaxWallet() public view returns (uint256) {
498   return _maxWalletSize / (10**_decimals);
499   }
500
501   function setSwapSettings(uint256 thresholdPercent, uint256 thresholdDivisor,
uint256 amountPercent, uint256 amountDivisor) external onlyOwner {
502
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 498

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
497   function getMaxWallet() public view returns (uint256) {
498   return _maxWalletSize / (10**_decimals);
499   }
500
501   function setSwapSettings(uint256 thresholdPercent, uint256 thresholdDivisor,
uint256 amountPercent, uint256 amountDivisor) external onlyOwner {
502
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 502

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
501   function setSwapSettings(uint256 thresholdPercent, uint256 thresholdDivisor,
uint256 amountPercent, uint256 amountDivisor) external onlyOwner {
502   swapThreshold = (_tTotal * thresholdPercent) / thresholdDivisor;
503   swapAmount = (_tTotal * amountPercent) / amountDivisor;
504   require(swapThreshold <= swapAmount, "Threshold cannot be above amount.");
505   }
506
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 502

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
501   function setSwapSettings(uint256 thresholdPercent, uint256 thresholdDivisor,
uint256 amountPercent, uint256 amountDivisor) external onlyOwner {
502   swapThreshold = (_tTotal * thresholdPercent) / thresholdDivisor;
503   swapAmount = (_tTotal * amountPercent) / amountDivisor;
504   require(swapThreshold <= swapAmount, "Threshold cannot be above amount.");
505   }
506
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 503

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
502   swapThreshold = (_tTotal * thresholdPercent) / thresholdDivisor;
503   swapAmount = (_tTotal * amountPercent) / amountDivisor;
504   require(swapThreshold <= swapAmount, "Threshold cannot be above amount.");
505   }
506
507
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 503

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
502   swapThreshold = (_tTotal * thresholdPercent) / thresholdDivisor;
503   swapAmount = (_tTotal * amountPercent) / amountDivisor;
504   require(swapThreshold <= swapAmount, "Threshold cannot be above amount.");
505   }
506
507
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 520

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
519   function setRewardsProperties(uint256 _minPeriod, uint256 _minReflection, uint256
minReflectionMultiplier) external onlyOwner {
520   _minReflection = _minReflection * 10**minReflectionMultiplier;
521   reflector.setRewardsProperties(_minPeriod, _minReflection);
522   }
523
524
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 520

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
519   function setRewardsProperties(uint256 _minPeriod, uint256 _minReflection, uint256
minReflectionMultiplier) external onlyOwner {
520   _minReflection = _minReflection * 10**minReflectionMultiplier;
521   reflector.setRewardsProperties(_minPeriod, _minReflection);
522   }
523
524
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 570

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
569   if (!_isExcludedFromLimits[to]) {
570   require(balanceOf(to) + amount <= _maxWalletSize, "Transfer amount exceeds the
maxWalletSize.");
571   }
572   }
573   }
574
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 582

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
581   uint256 swapAmt = swapAmount;
582   if(piContractSwapsEnabled) { swapAmt = (balanceOf(lpPair) * piSwapPercent) /
masterTaxDivisor; }
583   if(contractTokenBalance >= swapAmt) { contractTokenBalance = swapAmt; }
584   contractSwap(contractTokenBalance);
585   }
586
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 582

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
581   uint256 swapAmt = swapAmount;
582   if(piContractSwapsEnabled) { swapAmt = (balanceOf(lpPair) * piSwapPercent) /
masterTaxDivisor; }
583   if(contractTokenBalance >= swapAmt) { contractTokenBalance = swapAmt; }
584   contractSwap(contractTokenBalance);
585   }
586
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 618

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
617
618   _tOwned[from] -= amount;
619   uint256 amountReceived = amount;
620   if (takeFee) {
621   amountReceived = takeTaxes(from, amount, buy, sell, other);
622
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 623

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
622   }
623   _tOwned[to] += amountReceived;
624
625   processRewards(from, to);
626
627
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 657

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
656
657   uint256 feeAmount = amount * currentFee / masterTaxDivisor;
658
659   _tOwned[address(this)] += feeAmount;
660   emit Transfer(from, address(this), feeAmount);
661
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 657

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
656
657   uint256 feeAmount = amount * currentFee / masterTaxDivisor;
658
659   _tOwned[address(this)] += feeAmount;
660   emit Transfer(from, address(this), feeAmount);
661
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 659

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
658
659   _tOwned[address(this)] += feeAmount;
660   emit Transfer(from, address(this), feeAmount);
661
662   return amount - feeAmount;
663
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 662

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
661
662   return amount - feeAmount;
663   }
664
665   function contractSwap(uint256 contractTokenBalance) internal swapping {
666
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 675

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
674
675   uint256 toLiquify = ((contractTokenBalance * ratios.liquidity) / (ratios.total)) /
2;
676   uint256 swapAmt = contractTokenBalance - toLiquify;
677
678   address[] memory path = new address[](2);
679
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 675

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
674
675   uint256 toLiquify = ((contractTokenBalance * ratios.liquidity) / (ratios.total)) /
2;
676   uint256 swapAmt = contractTokenBalance - toLiquify;
677
678   address[] memory path = new address[](2);
679
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 675

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
674
675   uint256 toLiquify = ((contractTokenBalance * ratios.liquidity) / (ratios.total)) /
2;
676   uint256 swapAmt = contractTokenBalance - toLiquify;
677
678   address[] memory path = new address[](2);
679
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 676

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
675   uint256 toLiquify = ((contractTokenBalance * ratios.liquidity) / (ratios.total)) /
2;
676   uint256 swapAmt = contractTokenBalance - toLiquify;
677
678   address[] memory path = new address[](2);
679   path[0] = address(this);
680
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 691

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
690   uint256 amtBalance = address(this).balance;
691   uint256 liquidityBalance = (amtBalance * toLiquify) / swapAmt;
692
693   if (toLiquify > 0) {
694   dexRouter.addLiquidityETH{value: liquidityBalance}(
695
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 691

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
690   uint256 amtBalance = address(this).balance;
691   uint256 liquidityBalance = (amtBalance * toLiquify) / swapAmt;
692
693   if (toLiquify > 0) {
694   dexRouter.addLiquidityETH{value: liquidityBalance}(
695
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 705

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
704
705   amtBalance -= liquidityBalance;
706   ratios.total -= ratios.liquidity;
707   bool success;
708   uint256 rewardsBalance = (amtBalance * ratios.rewards) / ratios.total;
709
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 706

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
705   amtBalance -= liquidityBalance;
706   ratios.total -= ratios.liquidity;
707   bool success;
708   uint256 rewardsBalance = (amtBalance * ratios.rewards) / ratios.total;
709   uint256 marketingBalance = amtBalance - (rewardsBalance);
710
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 708

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
707   bool success;
708   uint256 rewardsBalance = (amtBalance * ratios.rewards) / ratios.total;
709   uint256 marketingBalance = amtBalance - (rewardsBalance);
710
711   if (ratios.rewards > 0) {
712
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 708

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
707   bool success;
708   uint256 rewardsBalance = (amtBalance * ratios.rewards) / ratios.total;
709   uint256 marketingBalance = amtBalance - (rewardsBalance);
710
711   if (ratios.rewards > 0) {
712
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 709

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
708   uint256 rewardsBalance = (amtBalance * ratios.rewards) / ratios.total;
709   uint256 marketingBalance = amtBalance - (rewardsBalance);
710
711   if (ratios.rewards > 0) {
712   try reflector.load{value: rewardsBalance}() {} catch {}
713
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 738

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
737   require(accounts.length == amounts.length, "Lengths do not match.");
738   for (uint8 i = 0; i < accounts.length; i++) {
739   require(balanceOf(msg.sender) >= amounts[i]);
740   _finalizeTransfer(msg.sender, accounts[i], amounts[i]*10**_decimals, false, false,
true);
741   }
742
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 740

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
739   require(balanceOf(msg.sender) >= amounts[i]);
740   _finalizeTransfer(msg.sender, accounts[i], amounts[i]*10**_decimals, false, false,
true);
741   }
742   }
743
744
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 740

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
739   require(balanceOf(msg.sender) >= amounts[i]);
740   _finalizeTransfer(msg.sender, accounts[i], amounts[i]*10**_decimals, false, false,
true);
741   }
742   }
743
744
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 6

low SEVERITY

The current pragma Solidity directive is ""=0.6.0<0.9.0"". 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

- AnimeVerse.sol

Locations

```
5   // SPDX-License-Identifier: MIT
6   pragma solidity >=0.6.0 <0.9.0;
7
8   interface IERC20 {
9   function totalSupply() external view returns (uint256);
10
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 126

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
125
126   mapping (address => uint256) _tOwned;
127   mapping (address => bool) lpPairs;
128   uint256 private timeSinceLastPair = 0;
129   mapping (address => mapping (address => uint256)) _allowances;
130
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 127

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
126   mapping (address => uint256) _tOwned;
127   mapping (address => bool) lpPairs;
128   uint256 private timeSinceLastPair = 0;
129   mapping (address => mapping (address => uint256)) _allowances;
130   mapping (address => bool) private _isExcludedFromProtection;
131
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 129

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

- AnimeVerse.sol

Locations

```
128   uint256 private timeSinceLastPair = 0;
129   mapping (address => mapping (address => uint256)) _allowances;
130   mapping (address => bool) private _isExcludedFromProtection;
131   mapping (address => bool) private _isExcludedFromFees;
132   mapping (address => bool) private _isExcludedFromLimits;
133
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 193

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
192
193   Cashier reflector;
194   uint256 reflectorGas = 300000;
195
196   bool inSwap;
197
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 194

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
193   Cashier reflector;
194   uint256 reflectorGas = 300000;
195
196   bool inSwap;
197   bool public contractSwapEnabled = false;
198
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 196

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
195
196   bool inSwap;
197   bool public contractSwapEnabled = false;
198   uint256 public swapThreshold;
199   uint256 public swapAmount;
200
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 207

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
206   bool public _hasLiqBeenAdded = false;
207   AntiSnipe antiSnipe;
208
209   modifier swapping() {
210   inSwap = true;
211
```

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

LINE 532

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
531   && to != _owner
532   && tx.origin != _owner
533   && !_liquidityHolders[to]
534   && !_liquidityHolders[from]
535   && to != DEAD
536
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 419

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
418   for(uint256 i = 0; i < accounts.length; i++){
419   setDividendExcluded(accounts[i], enabled);
420   }
421   }
422
423
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 679

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
678   address[] memory path = new address[](2);
679   path[0] = address(this);
680   path[1] = dexRouter.WETH();
681
682   dexRouter.swapExactTokensForETHSupportingFeeOnTransferTokens(
683
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 680

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
679   path[0] = address(this);
680   path[1] = dexRouter.WETH();
681
682   dexRouter.swapExactTokensForETHSupportingFeeOnTransferTokens(
683   swapAmt,
684
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 739

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
738   for (uint8 i = 0; i < accounts.length; i++) {
739   require(balanceOf(msg.sender) >= amounts[i]);
740   _finalizeTransfer(msg.sender, accounts[i], amounts[i]*10**_decimals, false, false,
true);
741   }
742   }
743
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 740

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
739   require(balanceOf(msg.sender) >= amounts[i]);
740   _finalizeTransfer(msg.sender, accounts[i], amounts[i]*10**_decimals, false, false,
true);
741   }
742   }
743
744
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 740

low SEVERITY

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

Source File

- AnimeVerse.sol

Locations

```
739   require(balanceOf(msg.sender) >= amounts[i]);
740   _finalizeTransfer(msg.sender, accounts[i], amounts[i]*10**_decimals, false, false,
true);
741   }
742   }
743
744
```

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

LINE 446

low SEVERITY

The environment variable "block.number" looks like it might be used as a source of randomness. Note that the values of variables like coinbase, gaslimit, block number and timestamp are predictable and can be manipulated by a malicious miner. Also keep in mind that attackers know hashes of earlier blocks. Don't use any of those environment variables as sources of randomness and be aware that use of these variables introduces a certain level of trust into miners.

Source File

- AnimeVerse.sol

Locations

```
445   }
446   try antiSnipe.setLaunch(lpPair, uint32(block.number), uint64(block.timestamp),
_decimals) {} catch {}
447   try reflector.initialize() {} catch {}
448   tradingEnabled = true;
449   swapThreshold = (balanceOf(lpPair) * 10) / 10000;
450
```

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.