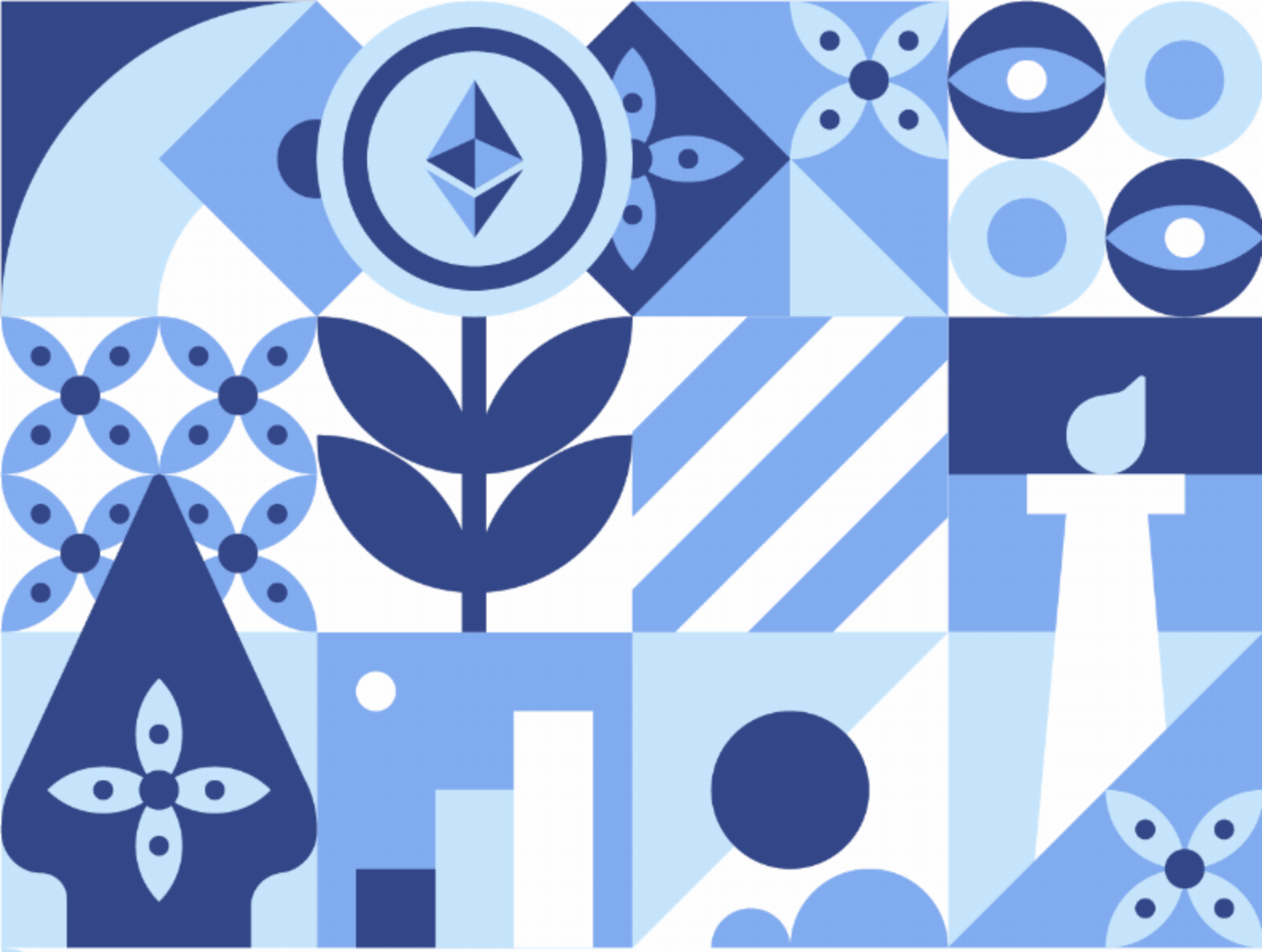

Retrogression

Smart Contract Audit Report

SYSFIXED

15 Feb 2022

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
Retrogression	RTGN	Ethereum

Addresses

Contract address	0x304243A820D4A3718BecC89a3F33513586162CF0
Contract deployer address	0x387A76e6642385778Cbc465FA8bf310045F41b25

Project Website

https://retrogression.filmcrib.io/

Codebase

https://etherscan.io/address/0x304243A820D4A3718BecC89a3F33513586162CF0#code

SUMMARY

Welcome to the FILM CRI3 and Retrogression Film, NFT and gaming Studio. Buy the tokens, play the game, collect the NFT's, watch the streamer.

Contract Summary

Documentation Quality

Retrogression 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 Retrogression 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 180, 199, 221, 254, 256, 277, 278, 303, 305, 635, 647, 930, 930, 932, 933, 939, 941, 945, 945, 947, 951, 973, 980, 985, 986, 992, 997, 1001, 1008, 1013, 1014, 1020, 1025, 1030, 1035, 1040, 1148, 1148, 1149, 1149, 1150, 1150, 1151, 1151, 1184, 1184, 1192, 1192, 1287, 1487, 1487, 1495, 1495, 1496, 1496, 1529 and 1529.
- 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 1402 and 1403.

CONCLUSION

We have audited the Retrogression project released on February-2022 to discover issues and identify potential security vulnerabilities in Retrogression 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 Retrogression 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. It is recommended to use vetted safe math libraries for arithmetic operations consistently.

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.	PASS
Unchecked Call Return Value	SWC-104	The return value of a message call should be checked.	PASS
Unprotected Ether Withdrawal	SWC-105	Due to missing or insufficient access controls, malicious parties can withdraw from the contract.	PASS
SELFDESTRUCT Instruction	SWC-106	The contract should not be self-destructible while it has funds belonging to users.	PASS
Reentrancy	SWC-107	Check effect interaction pattern should be followed if the code performs recursive call.	PASS
Uninitialized Storage Pointer	SWC-109	Uninitialized local storage variables can point to unexpected storage locations in the contract.	PASS
Assert Violation	SWC-110 SWC-123	Properly functioning code should never reach a failing assert statement.	ISSUE FOUND
Deprecated Solidity Functions	SWC-111	Deprecated built-in functions should never be used.	PASS
Delegate call to Untrusted Callee	SWC-112	Delegatecalls should only be allowed to trusted addresses.	PASS

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

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

SMART CONTRACT ANALYSIS

Started	Monday Feb 14 2022 01:04:17 GMT+0000 (Coordinated Universal Time)
Finished	Tuesday Feb 15 2022 12:08:51 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	Retrogression.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-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 180

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
179   unchecked {
180   _approve(sender, _msgSender(), currentAllowance - amount);
181   }
182
183   return true;
184
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 199

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
198   function increaseAllowance(address spender, uint256 addedValue) public virtual
returns (bool) {
199   _approve(_msgSender(), spender, _allowances[_msgSender()][spender] + addedValue);
200   return true;
201   }
202
203
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 221

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
220   unchecked {
221   _approve(_msgSender(), spender, currentAllowance - subtractedValue);
222   }
223
224   return true;
225
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 254

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
253   unchecked {
254   _balances[sender] = senderBalance - amount;
255   }
256   _balances[recipient] += amount;
257
258
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 256

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
255   }
256   _balances[recipient] += amount;
257
258   emit Transfer(sender, recipient, amount);
259
260
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 277

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
276
277   _totalSupply += amount;
278   _balances[account] += amount;
279   emit Transfer(address(0), account, amount);
280
281
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 278

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
277   _totalSupply += amount;
278   _balances[account] += amount;
279   emit Transfer(address(0), account, amount);
280
281   _afterTokenTransfer(address(0), account, amount);
282
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 303

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
302   unchecked {
303   _balances[account] = accountBalance - amount;
304   }
305   _totalSupply -= amount;
306
307
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 305

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
304   }
305   _totalSupply -= amount;
306
307   emit Transfer(account, address(0), amount);
308
309
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 635

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
634   ) internal {
635   uint256 newAllowance = token.allowance(address(this), spender) + value;
636   _callOptionalReturn(token, abi.encodeWithSelector(token.approve.selector, spender,
newAllowance));
637   }
638
639
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 647

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
646   require(oldAllowance >= value, "SafeERC20: decreased allowance below zero");
647   uint256 newAllowance = oldAllowance - value;
648   _callOptionalReturn(token, abi.encodeWithSelector(token.approve.selector, spender,
newAllowance));
649   }
650   }
651
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 930

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
929
930   require(!(a == - 2**255 && b == -1) && !(b == - 2**255 && a == -1));
931
932   int256 c = a * b;
933   require((b == 0) || (c / b == a));
934
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 930

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
929
930   require(!(a == - 2**255 && b == -1) && !(b == - 2**255 && a == -1));
931
932   int256 c = a * b;
933   require((b == 0) || (c / b == a));
934
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 932

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
931
932   int256 c = a * b;
933   require((b == 0) || (c / b == a));
934   return c;
935   }
936
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 933

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
932   int256 c = a * b;
933   require((b == 0) || (c / b == a));
934   return c;
935   }
936
937
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 939

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
938
939   require(!(a == - 2**255 && b == -1) && (b > 0));
940
941   return a / b;
942   }
943
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 941

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
940
941   return a / b;
942   }
943
944   function sub(int256 a, int256 b) internal pure returns (int256) {
945
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 945

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
944   function sub(int256 a, int256 b) internal pure returns (int256) {
945   require((b >= 0 && a - b <= a) || (b < 0 && a - b > a));
946
947   return a - b;
948   }
949
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 945

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
944   function sub(int256 a, int256 b) internal pure returns (int256) {
945   require((b >= 0 && a - b <= a) || (b < 0 && a - b > a));
946
947   return a - b;
948   }
949
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 947

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
946
947   return a - b;
948   }
949
950   function add(int256 a, int256 b) internal pure returns (int256) {
951
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 951

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
950   function add(int256 a, int256 b) internal pure returns (int256) {
951   int256 c = a + b;
952   require((b >= 0 && c >= a) || (b < 0 && c < a));
953   return c;
954   }
955
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 973

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
972   function tryAdd(uint256 a, uint256 b) internal pure returns (bool, uint256) {
973   uint256 c = a + b;
974   if (c < a) return (false, 0);
975   return (true, c);
976   }
977
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 980

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
979   if (b > a) return (false, 0);
980   return (true, a - b);
981   }
982
983   function tryMul(uint256 a, uint256 b) internal pure returns (bool, uint256) {
984
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 985

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
984   if (a == 0) return (true, 0);
985   uint256 c = a * b;
986   if (c / a != b) return (false, 0);
987   return (true, c);
988   }
989
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 986

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
985   uint256 c = a * b;
986   if (c / a != b) return (false, 0);
987   return (true, c);
988   }
989
990
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 992

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
991   if (b == 0) return (false, 0);
992   return (true, a / b);
993   }
994
995   function tryMod(uint256 a, uint256 b) internal pure returns (bool, uint256) {
996
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 997

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
996   if (b == 0) return (false, 0);
997   return (true, a % b);
998   }
999
1000   function add(uint256 a, uint256 b) internal pure returns (uint256) {
1001
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1001

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1000   function add(uint256 a, uint256 b) internal pure returns (uint256) {
1001   uint256 c = a + b;
1002   require(c >= a, "SafeMath: addition overflow");
1003   return c;
1004   }
1005
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1008

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1007   require(b <= a, "SafeMath: subtraction overflow");
1008   return a - b;
1009   }
1010
1011   function mul(uint256 a, uint256 b) internal pure returns (uint256) {
1012
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1013

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1012   if (a == 0) return 0;
1013   uint256 c = a * b;
1014   require(c / a == b, "SafeMath: multiplication overflow");
1015   return c;
1016   }
1017
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1014

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1013   uint256 c = a * b;
1014   require(c / a == b, "SafeMath: multiplication overflow");
1015   return c;
1016   }
1017
1018
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1020

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1019   require(b > 0, "SafeMath: division by zero");
1020   return a / b;
1021   }
1022
1023   function mod(uint256 a, uint256 b) internal pure returns (uint256) {
1024
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 1025

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1024   require(b > 0, "SafeMath: modulo by zero");
1025   return a % b;
1026   }
1027
1028   function sub(uint256 a, uint256 b, string memory errorMessage) internal pure
returns (uint256) {
1029
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1030

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1029   require(b <= a, errorMessage);
1030   return a - b;
1031   }
1032
1033   function div(uint256 a, uint256 b, string memory errorMessage) internal pure
returns (uint256) {
1034
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1035

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1034   require(b > 0, errorMessage);
1035   return a / b;
1036   }
1037
1038   function mod(uint256 a, uint256 b, string memory errorMessage) internal pure
returns (uint256) {
1039
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 1040

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1039   require(b > 0, errorMessage);
1040   return a % b;
1041   }
1042   }
1043
1044
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1148

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1147
1148   swapTokensAtAmount = 2500000 * (10**18);
1149   _maxWallet = 5000000 * (10**18);
1150   _maxBuy = 2500000 * (10**18);
1151   _maxSell = 2500000 * (10**18);
1152
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1148

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1147
1148   swapTokensAtAmount = 2500000 * (10**18);
1149   _maxWallet = 5000000 * (10**18);
1150   _maxBuy = 2500000 * (10**18);
1151   _maxSell = 2500000 * (10**18);
1152
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1149

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1148   swapTokensAtAmount = 2500000 * (10**18);
1149   _maxWallet = 5000000 * (10**18);
1150   _maxBuy = 2500000 * (10**18);
1151   _maxSell = 2500000 * (10**18);
1152
1153
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1149

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1148   swapTokensAtAmount = 2500000 * (10**18);
1149   _maxWallet = 5000000 * (10**18);
1150   _maxBuy = 2500000 * (10**18);
1151   _maxSell = 2500000 * (10**18);
1152
1153
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1150

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1149   _maxWallet = 5000000 * (10**18);
1150   _maxBuy = 2500000 * (10**18);
1151   _maxSell = 2500000 * (10**18);
1152
1153   totalBuyFees = _buyProductionFee.add(_buyMarketingFee).add(_buyFilmCribFee);
1154
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1150

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1149   _maxWallet = 5000000 * (10**18);
1150   _maxBuy = 2500000 * (10**18);
1151   _maxSell = 2500000 * (10**18);
1152
1153   totalBuyFees = _buyProductionFee.add(_buyMarketingFee).add(_buyFilmCribFee);
1154
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1151

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1150   _maxBuy = 2500000 * (10**18);
1151   _maxSell = 2500000 * (10**18);
1152
1153   totalBuyFees = _buyProductionFee.add(_buyMarketingFee).add(_buyFilmCribFee);
1154   totalSellFees = _sellProductionFee.add(_sellMarketingFee).add(_sellFilmCribFee);
1155
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1151

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1150   _maxBuy = 2500000 * (10**18);
1151   _maxSell = 2500000 * (10**18);
1152
1153   totalBuyFees = _buyProductionFee.add(_buyMarketingFee).add(_buyFilmCribFee);
1154   totalSellFees = _sellProductionFee.add(_sellMarketingFee).add(_sellFilmCribFee);
1155
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1184

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1183   */
1184   _mint(owner(), 1000000000 * (10**18));
1185   }
1186
1187   receive() external payable {
1188
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1184

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1183   */
1184   _mint(owner(), 1000000000 * (10**18));
1185   }
1186
1187   receive() external payable {
1188
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1192

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1191   function updateSwapAmount(uint256 amount) public onlyOwner {
1192   swapTokensAtAmount = amount * (10**18);
1193
1194   emit UpdateSwapAmount(amount, swapTokensAtAmount);
1195   }
1196
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1192

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1191   function updateSwapAmount(uint256 amount) public onlyOwner {
1192   swapTokensAtAmount = amount * (10**18);
1193
1194   emit UpdateSwapAmount(amount, swapTokensAtAmount);
1195   }
1196
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1287

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1286   uint256 contractBalanceRecipient = balanceOf(to);
1287   require(contractBalanceRecipient + amount <= _maxWallet, "Exceeds maximum wallet
token amount.");
1288   }
1289
1290   if(amount == 0) {
1291
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1487

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1486   require(newMaxWallet >= 5000000, "Cannot lower max wallet below .5% of total
supply");
1487   _maxWallet = newMaxWallet * (10**18);
1488
1489   emit UpdateMaxWallet(newMaxWallet, _maxWallet);
1490   }
1491
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1487

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1486   require(newMaxWallet >= 5000000, "Cannot lower max wallet below .5% of total
supply");
1487   _maxWallet = newMaxWallet * (10**18);
1488
1489   emit UpdateMaxWallet(newMaxWallet, _maxWallet);
1490   }
1491
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1495

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1494   require(newMaxSell >= 2500000, "Cannot lower max sell below .25% of total
supply");
1495   _maxBuy = newMaxBuy * (10**18);
1496   _maxSell = newMaxSell * (10**18);
1497
1498   emit UpdateMaxBuySell(newMaxBuy, _maxBuy, newMaxSell, _maxSell);
1499
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1495

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1494   require(newMaxSell >= 2500000, "Cannot lower max sell below .25% of total
supply");
1495   _maxBuy = newMaxBuy * (10**18);
1496   _maxSell = newMaxSell * (10**18);
1497
1498   emit UpdateMaxBuySell(newMaxBuy, _maxBuy, newMaxSell, _maxSell);
1499
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1496

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1495   _maxBuy = newMaxBuy * (10**18);
1496   _maxSell = newMaxSell * (10**18);
1497
1498   emit UpdateMaxBuySell(newMaxBuy, _maxBuy, newMaxSell, _maxSell);
1499   }
1500
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1496

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1495   _maxBuy = newMaxBuy * (10**18);
1496   _maxSell = newMaxSell * (10**18);
1497
1498   emit UpdateMaxBuySell(newMaxBuy, _maxBuy, newMaxSell, _maxSell);
1499   }
1500
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1529

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1528   require(amount <= balanceOf(address(this)), "Amount cannot exceed tokens in
contract");
1529   super._transfer(address(this), 0x000000000000000000000000000000000000dEaD, amount
* 10**18);
1530   }
1531   }
1532
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1529

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1528   require(amount <= balanceOf(address(this)), "Amount cannot exceed tokens in
contract");
1529   super._transfer(address(this), 0x000000000000000000000000000000000000dEaD, amount
* 10**18);
1530   }
1531   }
1532
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1402

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1401   address[] memory path = new address[](2);
1402   path[0] = address(this);
1403   path[1] = uniswapV2Router.WETH();
1404
1405   _approve(address(this), address(uniswapV2Router), tokenAmount);
1406
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1403

low SEVERITY

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

Source File

- Retrogression.sol

Locations

```
1402   path[0] = address(this);
1403   path[1] = uniswapV2Router.WETH();
1404
1405   _approve(address(this), address(uniswapV2Router), tokenAmount);
1406
1407
```

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.