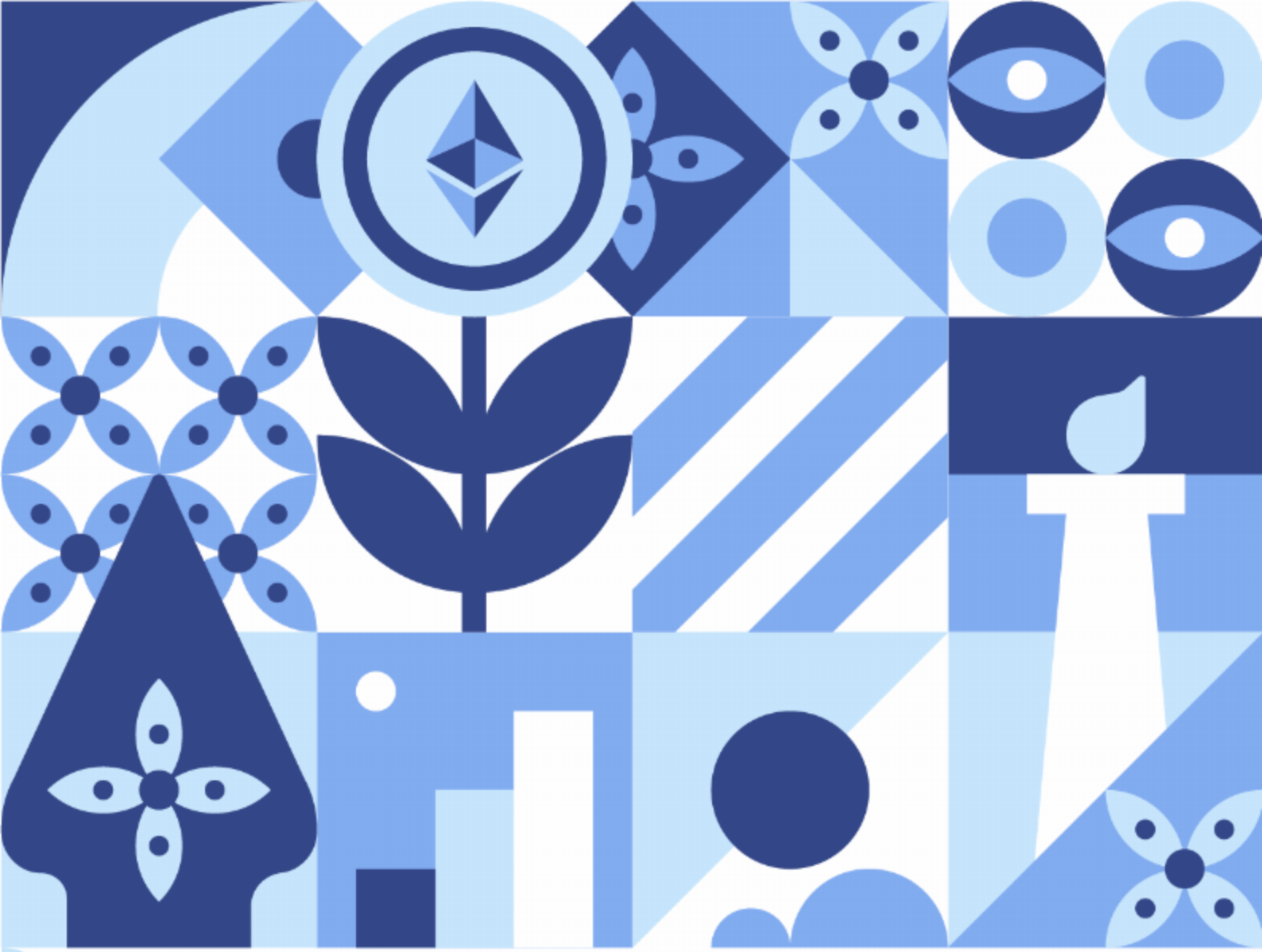

Mandox

Smart Contract Audit Report

SYSFIXED

05 Apr 2022

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
Mandox	MANDOX	Ethereum

Addresses

Contract address	0x33d203fa03bb30b133de0fe2d6533c268ba286b6
Contract deployer address	0x958D4277F9C049108256E7f8b61765575B6D92a9

Project Website

https://mandoxglobal.net/

Codebase

https://etherscan.io/address/0x33d203fa03bb30b133de0fe2d6533c268ba286b6#code

SUMMARY

Mandox LLC is a company in Wyoming, United States. We launched Mandox November 26th 2021. The development team is fully doxxed & KYC certified. Mandox is creating a forever growing ecosystem that bridges the gap between cryptocurrencies & NFTs. Mandox specializes in Crypto, NFTs, NFT & Token Staking, which allows investors to earn passive income. There is also a P2E Game in development, Mandox Play and Create NFT Marketplace.

Contract Summary

Documentation Quality

Mandox 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 Mandox 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 419 and 444.
- SWC-101 | It is recommended to use vetted safe math libraries for arithmetic operations consistently on lines 74, 83, 92, 93, 108, 115, 451, 451, 452, 452, 474, 474, 475, 475, 562, 563, 578, 578, 578, 605, 607, 657, 658, 666, 666, 736, 767, 771, 853 and 607.
- SWC-103 | Pragma statements can be allowed to float when a contract is intended on lines 42.
- 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 577, 578, 605, 606, 606, 738, 741, 742, 746, 864 and 865.

CONCLUSION

We have audited the Mandox project released on April 2022 to discover issues and identify potential security vulnerabilities in Mandox 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 Mandox 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 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.	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 Apr 04 2022 05:48:05 GMT+0000 (Coordinated Universal Time)
Finished	Tuesday Apr 05 2022 11:36:46 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	MandoxV2Token.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	COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-108	STATE VARIABLE VISIBILITY IS NOT SET.	low	acknowledged
SWC-108	STATE VARIABLE VISIBILITY IS NOT SET.	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 74

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
73
74   function div(uint256 a, uint256 b, string memory errorMessage) internal pure returns
(uint256) {
75   require(b > 0, errorMessage);
76   uint256 c = a / b;
77   // assert(a == b * c + a % b); // There is no case in which this doesn't hold
78
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 83

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
82   function mod(uint256 a, uint256 b) internal pure returns (uint256) {
83   return mod(a, b, "SafeMath: modulo by zero");
84   }
85
86   function mod(uint256 a, uint256 b, string memory errorMessage) internal pure returns
(uint256) {
87
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 92

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
91
92   abstract contract Context {
93
94   function _msgSender() internal view virtual returns (address) {
95   return msg.sender;
96
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 93

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
92   abstract contract Context {
93
94   function _msgSender() internal view virtual returns (address) {
95   return msg.sender;
96   }
97
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 108

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
107   bytes32 codehash;
108   bytes32 accountHash =
0xc5d2460186f7233c927e7db2dcc703c0e500b653ca82273b7bfad8045d85a470;
109   assembly { codehash := extcodehash(account) }
110   return (codehash != accountHash && codehash != 0x0);
111   }
112
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 115

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
114   require(address(this).balance >= amount, "Address: insufficient balance");
115   (bool success, ) = recipient.call{ value: amount }("");
116   require(success, "Address: unable to send value, recipient may have reverted");
117   }
118
119
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 451

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
450
451   event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);
452   event SwapAndLiquifyEnabledUpdated(bool enabled);
453   event SwapAndLiquify(
454   uint256 tokensSwapped,
455
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 451

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
450
451   event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);
452   event SwapAndLiquifyEnabledUpdated(bool enabled);
453   event SwapAndLiquify(
454   uint256 tokensSwapped,
455
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 452

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
451   event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);
452   event SwapAndLiquifyEnabledUpdated(bool enabled);
453   event SwapAndLiquify(
454   uint256 tokensSwapped,
455   uint256 ethReceived,
456
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 452

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
451   event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);
452   event SwapAndLiquifyEnabledUpdated(bool enabled);
453   event SwapAndLiquify(
454   uint256 tokensSwapped,
455   uint256 ethReceived,
456
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 474

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
473
474   _isExcludedFromFee[owner()] = true;
475   _isExcludedFromFee[address(this)] = true;
476
477   emit Transfer(address(0), _msgSender(), _tTotal);
478
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 474

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
473
474   _isExcludedFromFee[owner()] = true;
475   _isExcludedFromFee[address(this)] = true;
476
477   emit Transfer(address(0), _msgSender(), _tTotal);
478
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 475

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
474   _isExcludedFromFee[owner()] = true;
475   _isExcludedFromFee[address(this)] = true;
476
477   emit Transfer(address(0), _msgSender(), _tTotal);
478   }
479
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 475

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
474   _isExcludedFromFee[owner()] = true;
475   _isExcludedFromFee[address(this)] = true;
476
477   emit Transfer(address(0), _msgSender(), _tTotal);
478   }
479
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 562

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
561   require(tAmount <= _tTotal, "Amount must be less than supply");
562   if (!deductTransferFee) {
563   (uint256 rAmount,,,,,) = _getValues(tAmount);
564   return rAmount;
565   } else {
566
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 563

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
562   if (!deductTransferFee) {
563   (uint256 rAmount,,,,,) = _getValues(tAmount);
564   return rAmount;
565   } else {
566   (,uint256 rTransferAmount,,,,) = _getValues(tAmount);
567
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 578

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
577   function excludeFromReward(address account) public onlyOwner() {
578   require(!_isExcluded[account], "Account is already excluded");
579   if(_rOwned[account] > 0) {
580   _tOwned[account] = tokenFromReflection(_rOwned[account]);
581   }
582
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 578

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
577   function excludeFromReward(address account) public onlyOwner() {
578   require(!_isExcluded[account], "Account is already excluded");
579   if(_rOwned[account] > 0) {
580   _tOwned[account] = tokenFromReflection(_rOwned[account]);
581   }
582
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 578

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
577   function excludeFromReward(address account) public onlyOwner() {
578   require(!_isExcluded[account], "Account is already excluded");
579   if(_rOwned[account] > 0) {
580   _tOwned[account] = tokenFromReflection(_rOwned[account]);
581   }
582
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 605

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
604   _takeLiquidity(tLiquidity);
605   _reflectFee(rFee, tFee);
606   emit Transfer(sender, recipient, tTransferAmount);
607   }
608
609
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 607

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
606   emit Transfer(sender, recipient, tTransferAmount);
607   }
608
609   function excludeFromFee(address account) public onlyOwner {
610   _isExcludedFromFee[account] = true;
611
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 657

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
656
657   function addBotWallet(address botwallet) external onlyOwner() {
658   botWallets[botwallet] = true;
659   }
660
661
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 658

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
657   function addBotWallet(address botwallet) external onlyOwner() {
658   botWallets[botwallet] = true;
659   }
660
661   function removeBotWallet(address botwallet) external onlyOwner() {
662
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 666

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
665   function getBotWalletStatus(address botwallet) public view returns (bool) {
666   return botWallets[botwallet];
667   }
668
669   function allowtrading()external onlyOwner() {
670
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 666

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
665   function getBotWalletStatus(address botwallet) public view returns (bool) {
666   return botWallets[botwallet];
667   }
668
669   function allowtrading()external onlyOwner() {
670
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 736

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
735   function calculateTaxFee(uint256 _amount) private view returns (uint256) {
736   return _amount.mul(_taxFee).div(
737   10**2
738   );
739   }
740
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 767

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
766   function _approve(address owner, address spender, uint256 amount) private {
767   require(owner != address(0), "ERC20: approve from the zero address");
768   require(spender != address(0), "ERC20: approve to the zero address");
769
770   _allowances[owner][spender] = amount;
771
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 771

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
770   _allowances[owner][spender] = amount;
771   emit Approval(owner, spender, amount);
772   }
773
774   function _transfer(
775
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 853

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
852
853   uniswapV2Router.addLiquidityETH{value: ethAmount}(
854   address(this),
855   tokenAmount,
856   0,
857
```

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

LINE 607

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
606   emit Transfer(sender, recipient, tTransferAmount);
607   }
608
609   function excludeFromFee(address account) public onlyOwner {
610   _isExcludedFromFee[account] = true;
611
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 42

low SEVERITY

The current pragma Solidity directive is ""^0.8.9"". 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

- MandoxV2Token.sol

Locations

```
41   uint256 c = a + b;
42   require(c >= a, "SafeMath: addition overflow");
43
44   return c;
45   }
46
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 419

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
418   mapping (address => bool) private botWallets;
419   bool botscantrade = false;
420
421   bool public canTrade = false;
422
423
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 444

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
443
444   bool inSwapAndLiquify;
445   bool public swapAndLiquifyEnabled = true;
446
447   uint256 public _maxTxAmount = 10000000000 * 10**9;
448
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 577

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
576
577   function excludeFromReward(address account) public onlyOwner() {
578   require(!_isExcluded[account], "Account is already excluded");
579   if(_rOwned[account] > 0) {
580   _tOwned[account] = tokenFromReflection(_rOwned[account]);
581
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 578

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
577   function excludeFromReward(address account) public onlyOwner() {
578   require(!_isExcluded[account], "Account is already excluded");
579   if(_rOwned[account] > 0) {
580   _tOwned[account] = tokenFromReflection(_rOwned[account]);
581   }
582
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 605

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
604   _takeLiquidity(tLiquidity);
605   _reflectFee(rFee, tFee);
606   emit Transfer(sender, recipient, tTransferAmount);
607   }
608
609
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 606

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
605   _reflectFee(rFee, tFee);
606   emit Transfer(sender, recipient, tTransferAmount);
607   }
608
609   function excludeFromFee(address account) public onlyOwner {
610
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 606

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
605   _reflectFee(rFee, tFee);
606   emit Transfer(sender, recipient, tTransferAmount);
607   }
608
609   function excludeFromFee(address account) public onlyOwner {
610
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 738

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
737   10**2
738   );
739   }
740
741   function calculateLiquidityFee(uint256 _amount) private view returns (uint256) {
742
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 741

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
740
741   function calculateLiquidityFee(uint256 _amount) private view returns (uint256) {
742   return _amount.mul(_liquidityFee).div(
743   10**2
744   );
745
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 742

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
741   function calculateLiquidityFee(uint256 _amount) private view returns (uint256) {
742   return _amount.mul(_liquidityFee).div(
743   10**2
744   );
745   }
746
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 746

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
745   }
746
747   function removeAllFee() private {
748   if(_taxFee == 0 && _liquidityFee == 0) return;
749
750
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 864

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
863   function _tokenTransfer(address sender, address recipient, uint256 amount,bool
takeFee) private {
864   if(!canTrade){
865   require(sender == owner());
866   }
867
868
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 865

low SEVERITY

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

Source File

- MandoxV2Token.sol

Locations

```
864   if(!canTrade){
865   require(sender == owner());
866   }
867
868   if(botWallets[sender] || botWallets[recipient]){
869
```

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.