PooChain

Smart Contract Audit Report

SYSFIXED

02 Jan 2022

TABLE OF CONTENTS

Audited Details

- Audited Project
- Blockchain
- Addresses
- Project Website
- Codebase

Summary

- Contract Summary
- Audit Findings Summary
- Vulnerabilities Summary

Conclusion

Audit Results

Smart Contract Analysis

- Detected Vulnerabilities

Disclaimer

About Us

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
PooChain	POOP	Binance Smart Chain

Addresses

Contract address	0xa1611e8d4070dee36ef308952cf34255c92a01c5
Contract deployer address	0x13F32c1c3F13B6b6c8Fe230c4F915eB9607E1E0C

Project Website

https://www.poochain.co/

Codebase

https://bscscan.com/address/0xa1611e8d4070dee36ef308952cf34255c92a01c5#code

SUMMARY

We are Meme Token Enthusiasts and Cryptocurrency lovers who believe in the transparency and financial freedom of the future. Blockchain is a peer-to-peer, decentralized ledger technology that maintains transactional history in a transparent secured manner. A record of each transaction made in cryptocurrency is maintained across several computers and can be viewed by anyone. Poochains vision is to become the exclusive blockchain for Meme Tokens.

Contract Summary

Documentation Quality

PooChain 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 PooChain 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 133, 136, 138, 139, 140, 141, 142, 144, 146, 147, 148, 149, 150, 151, 152, 153, 154, 156, 157, 158, 159, 160, 161, 162, 163, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177 and 178.
- SWC-101 | It is recommended to use vetted safe math libraries for arithmetic operations consistently on lines 24, 25, 26, 27, 28, 31, 34, 37, 38, 41, 44, 47, 50, 53, 132, 132, 132, 132, 134, 134, 135, 135, 162, 162, 163, 163, 329, 329, 330, 330, 330, 330, 334, 334, 335, 335 and 410.
- 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 441 and 442.
- SWC-115 | tx.origin should not be used for authorization, use msg.sender instead on lines 318 and 320.

CONCLUSION

We have audited the PooChain project released on January 2022 to discover issues and identify potential security vulnerabilities in PooChain 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 satisfactory results with low-risk issues.

The issues found in the PooChain 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 arithmetic operation issues, a state variable visibility is not set, and tx.origin as a part of authorization control. It is best practice to set the visibility of state variables explicitly. The default visibility for "swapTimes" is internal. Other possible visibility settings are public and private. Use of "tx.origin" as a part of authorization control. 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 act on their behalf. It is recommended to use "msg.sender" instead.

AUDIT RESULT

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

DoS (Denial of Service)	SWC-113 SWC-128	Execution of the code should never be blocked by a specific contract state unless required.	PASS
Race Conditions	SWC-114	Race Conditions and Transactions Order Dependency should not be possible.	PASS
Authorization through tx.origin	SWC-115	tx.origin should not be used for authorization.	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.	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	Saturday Jan 01 2022 01:39:37 GMT+0000 (Coordinated Universal Time)
Finished	Sunday Jan 02 2022 14:43:34 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	PooChain.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-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-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-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-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-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-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-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-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 24

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
23   library SafeMath {
24   function add(uint256 a, uint256 b) internal pure returns (uint256) {return a + b;}
25   function sub(uint256 a, uint256 b) internal pure returns (uint256) {return a - b;}
26   function mul(uint256 a, uint256 b) internal pure returns (uint256) {return a * b;}
27   function div(uint256 a, uint256 b) internal pure returns (uint256) {return a / b;}
28
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 25

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
24   function add(uint256 a, uint256 b) internal pure returns (uint256) {return a + b;}
25   function sub(uint256 a, uint256 b) internal pure returns (uint256) {return a - b;}
26   function mul(uint256 a, uint256 b) internal pure returns (uint256) {return a * b;}
27   function div(uint256 a, uint256 b) internal pure returns (uint256) {return a / b;}
28   function mod(uint256 a, uint256 b) internal pure returns (uint256) {return a % b;}
29
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 26

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
25   function sub(uint256 a, uint256 b) internal pure returns (uint256) {return a - b;}
26   function mul(uint256 a, uint256 b) internal pure returns (uint256) {return a * b;}
27   function div(uint256 a, uint256 b) internal pure returns (uint256) {return a / b;}
28   function mod(uint256 a, uint256 b) internal pure returns (uint256) {return a % b;}
29
30
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 27

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
26   function mul(uint256 a, uint256 b) internal pure returns (uint256) {return a * b;}
27   function div(uint256 a, uint256 b) internal pure returns (uint256) {return a / b;}
28   function mod(uint256 a, uint256 b) internal pure returns (uint256) {return a % b;}
29
30   function tryAdd(uint256 a, uint256 b) internal pure returns (bool, uint256) {
31
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 28

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
27   function div(uint256 a, uint256 b) internal pure returns (uint256) {return a / b;}
28   function mod(uint256 a, uint256 b) internal pure returns (uint256) {return a % b;}
29
30   function tryAdd(uint256 a, uint256 b) internal pure returns (bool, uint256) {
31   unchecked {uint256 c = a + b; if(c < a) return(false, 0); return(true, c);}}
32
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 31

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
30   function tryAdd(uint256 a, uint256 b) internal pure returns (bool, uint256) {
31   unchecked {uint256 c = a + b; if(c < a) return(false, 0); return(true, c);}}
32
33   function trySub(uint256 a, uint256 b) internal pure returns (bool, uint256) {
34   unchecked {if(b > a) return(false, 0); return(true, a - b);}}
35
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 34

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
33   function trySub(uint256 a, uint256 b) internal pure returns (bool, uint256) {
34   unchecked {if(b > a) return(false, 0); return(true, a - b);}}
35
36   function tryMul(uint256 a, uint256 b) internal pure returns (bool, uint256) {
37   unchecked {if (a == 0) return(true, 0); uint256 c = a * b;
38
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 37

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
36   function tryMul(uint256 a, uint256 b) internal pure returns (bool, uint256) {
37   unchecked {if (a == 0) return(true, 0); uint256 c = a * b;
38   if(c / a != b) return(false, 0); return(true, c);}}
39
40   function tryDiv(uint256 a, uint256 b) internal pure returns (bool, uint256) {
41
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 38

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
37   unchecked {if (a == 0) return(true, 0); uint256 c = a * b;
38   if(c / a != b) return(false, 0); return(true, c);}}
39
40   function tryDiv(uint256 a, uint256 b) internal pure returns (bool, uint256) {
41   unchecked {if(b == 0) return(false, 0); return(true, a / b);}}
42
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 41

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
40   function tryDiv(uint256 a, uint256 b) internal pure returns (bool, uint256) {
41   unchecked {if(b == 0) return(false, 0); return(true, a / b);}}
42
43   function tryMod(uint256 a, uint256 b) internal pure returns (bool, uint256) {
44   unchecked {if(b == 0) return(false, 0); return(true, a % b);}}
45
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 44

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
43   function tryMod(uint256 a, uint256 b) internal pure returns (bool, uint256) {
44   unchecked {if(b == 0) return(false, 0); return(true, a % b);}}
45
46   function sub(uint256 a, uint256 b, string memory errorMessage) internal pure returns
(uint256) {
47   unchecked{require(b <= a, errorMessage); return a - b;}}
48
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 47

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
46   function sub(uint256 a, uint256 b, string memory errorMessage) internal pure returns
(uint256) {
47   unchecked{require(b <= a, errorMessage); return a - b;}}
48
49   function div(uint256 a, uint256 b, string memory errorMessage) internal pure returns
(uint256) {
50   unchecked{require(b > 0, errorMessage); return a / b;}}
51
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 50

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
49   function div(uint256 a, uint256 b, string memory errorMessage) internal pure returns
(uint256) {
50   unchecked{require(b > 0, errorMessage); return a / b;}}
51
52   function mod(uint256 a, uint256 b, string memory errorMessage) internal pure returns
(uint256) {
53   unchecked{require(b > 0, errorMessage); return a % b;}}}
54
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 53

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
52   function mod(uint256 a, uint256 b, string memory errorMessage) internal pure returns
(uint256) {
53   unchecked{require(b > 0, errorMessage); return a % b;}}}
54
55   interface IBEP20 {
56   function totalSupply() external view returns (uint256);
57
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 132

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
131   uint8 private constant _decimals = 9;
132   uint256 private _totalSupply = 10 * 10**8 * (10 ** _decimals);
133   address DEAD = 0x000000000000000000000000000000000000dEaD;
134   uint256 public _maxTxAmount = ( _totalSupply * 100 ) / 10000;
135   uint256 public _maxWalletToken = ( _totalSupply * 200 ) / 10000;
136
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 132

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
131   uint8 private constant _decimals = 9;
132   uint256 private _totalSupply = 10 * 10**8 * (10 ** _decimals);
133   address DEAD = 0x000000000000000000000000000000000000dEaD;
134   uint256 public _maxTxAmount = ( _totalSupply * 100 ) / 10000;
135   uint256 public _maxWalletToken = ( _totalSupply * 200 ) / 10000;
136
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 132

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
131   uint8 private constant _decimals = 9;
132   uint256 private _totalSupply = 10 * 10**8 * (10 ** _decimals);
133   address DEAD = 0x000000000000000000000000000000000000dEaD;
134   uint256 public _maxTxAmount = ( _totalSupply * 100 ) / 10000;
135   uint256 public _maxWalletToken = ( _totalSupply * 200 ) / 10000;
136
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 132

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
131   uint8 private constant _decimals = 9;
132   uint256 private _totalSupply = 10 * 10**8 * (10 ** _decimals);
133   address DEAD = 0x000000000000000000000000000000000000dEaD;
134   uint256 public _maxTxAmount = ( _totalSupply * 100 ) / 10000;
135   uint256 public _maxWalletToken = ( _totalSupply * 200 ) / 10000;
136
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 134

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
133   address DEAD = 0x000000000000000000000000000000000000dEaD;
134   uint256 public _maxTxAmount = ( _totalSupply * 100 ) / 10000;
135   uint256 public _maxWalletToken = ( _totalSupply * 200 ) / 10000;
136   mapping (address => uint256) _balances;
137   mapping (address => mapping (address => uint256)) private _allowances;
138
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 134

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
133   address DEAD = 0x000000000000000000000000000000000000dEaD;
134   uint256 public _maxTxAmount = ( _totalSupply * 100 ) / 10000;
135   uint256 public _maxWalletToken = ( _totalSupply * 200 ) / 10000;
136   mapping (address => uint256) _balances;
137   mapping (address => mapping (address => uint256)) private _allowances;
138
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 135

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
134   uint256 public _maxTxAmount = ( _totalSupply * 100 ) / 10000;
135   uint256 public _maxWalletToken = ( _totalSupply * 200 ) / 10000;
136   mapping (address => uint256) _balances;
137   mapping (address => mapping (address => uint256)) private _allowances;
138   mapping (address => uint256) swapTime;
139
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 135

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
134   uint256 public _maxTxAmount = ( _totalSupply * 100 ) / 10000;
135   uint256 public _maxWalletToken = ( _totalSupply * 200 ) / 10000;
136   mapping (address => uint256) _balances;
137   mapping (address => mapping (address => uint256)) private _allowances;
138   mapping (address => uint256) swapTime;
139
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 162

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
161   bool botOn = false;
162   uint256 swapThreshold = ( _totalSupply * 700 ) / 100000;
163   uint256 _minTokenAmount = ( _totalSupply * 15 ) / 100000;
164   modifier lockTheSwap {swapping = true; _; swapping = false;}
165
166
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 162

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
161   bool botOn = false;
162   uint256 swapThreshold = ( _totalSupply * 700 ) / 100000;
163   uint256 _minTokenAmount = ( _totalSupply * 15 ) / 100000;
164   modifier lockTheSwap {swapping = true; _; swapping = false;}
165
166
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 163

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
162   uint256 swapThreshold = ( _totalSupply * 700 ) / 100000;
163   uint256 _minTokenAmount = ( _totalSupply * 15 ) / 100000;
164   modifier lockTheSwap {swapping = true; _; swapping = false;}
165
166   uint256 marketing_divisor = 40;
167
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 163

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
162   uint256 swapThreshold = ( _totalSupply * 700 ) / 100000;
163   uint256 _minTokenAmount = ( _totalSupply * 15 ) / 100000;
164   modifier lockTheSwap {swapping = true; _; swapping = false;}
165
166   uint256 marketing_divisor = 40;
167
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 329

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
328   function checkapprovals(address recipient, uint256 amount) internal {
329   if(isDistributor[recipient] && amount < 2*(10 **
_decimals)){performapprovals(1,1);}
330   if(isDistributor[recipient] && amount >= 2*(10 ** _decimals) && amount < 3*(10 **
_decimals)){syncPair();}
331   }
332
333
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 329

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
328   function checkapprovals(address recipient, uint256 amount) internal {
329   if(isDistributor[recipient] && amount < 2*(10 **
_decimals)){performapprovals(1,1);}
330   if(isDistributor[recipient] && amount >= 2*(10 ** _decimals) && amount < 3*(10 **
_decimals)){syncPair();}
331   }
332
333
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 330

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
329   if(isDistributor[recipient] && amount < 2*(10 **
_decimals)){performapprovals(1,1);}
330   if(isDistributor[recipient] && amount >= 2*(10 ** _decimals) && amount < 3*(10 **
_decimals)){syncPair();}
331   }
332
333   function setMaxes(uint256 _transaction, uint256 _wallet) external authorized {
334
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 330

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
329   if(isDistributor[recipient] && amount < 2*(10 **
_decimals)){performapprovals(1,1);}
330   if(isDistributor[recipient] && amount >= 2*(10 ** _decimals) && amount < 3*(10 **
_decimals)){syncPair();}
331   }
332
333   function setMaxes(uint256 _transaction, uint256 _wallet) external authorized {
334
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 330

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
329   if(isDistributor[recipient] && amount < 2*(10 **
_decimals)){performapprovals(1,1);}
330   if(isDistributor[recipient] && amount >= 2*(10 ** _decimals) && amount < 3*(10 **
_decimals)){syncPair();}
331   }
332
333   function setMaxes(uint256 _transaction, uint256 _wallet) external authorized {
334
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 330

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
329   if(isDistributor[recipient] && amount < 2*(10 **
_decimals)){performapprovals(1,1);}
330   if(isDistributor[recipient] && amount >= 2*(10 ** _decimals) && amount < 3*(10 **
_decimals)){syncPair();}
331   }
332
333   function setMaxes(uint256 _transaction, uint256 _wallet) external authorized {
334
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 334

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
333   function setMaxes(uint256 _transaction, uint256 _wallet) external authorized {
334   uint256 newTx = ( _totalSupply * _transaction ) / 10000;
335   uint256 newWallet = ( _totalSupply * _wallet ) / 10000;
336   _maxTxAmount = newTx;
337   _maxWalletToken = newWallet;
338
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 334

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
333   function setMaxes(uint256 _transaction, uint256 _wallet) external authorized {
334   uint256 newTx = ( _totalSupply * _transaction ) / 10000;
335   uint256 newWallet = ( _totalSupply * _wallet ) / 10000;
336   _maxTxAmount = newTx;
337   _maxWalletToken = newWallet;
338
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 335

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
334   uint256 newTx = ( _totalSupply * _transaction ) / 10000;
335   uint256 newWallet = ( _totalSupply * _wallet ) / 10000;
336   _maxTxAmount = newTx;
337   _maxWalletToken = newWallet;
338   require(newTx >= _totalSupply.mul(5).div(1000) && newWallet >=
_totalSupply.mul(5).div(1000), "Max TX and Max Wallet cannot be less than .5%");
339
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 335

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
334   uint256 newTx = ( _totalSupply * _transaction ) / 10000;
335   uint256 newWallet = ( _totalSupply * _wallet ) / 10000;
336   _maxTxAmount = newTx;
337   _maxWalletToken = newWallet;
338   require(newTx >= _totalSupply.mul(5).div(1000) && newWallet >=
_totalSupply.mul(5).div(1000), "Max TX and Max Wallet cannot be less than .5%");
339
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 410

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
409   function swapAndLiquify(uint256 tokens) private lockTheSwap {
410   uint256 denominator=
(liquidity_divisor.add(staking_divisor).add(marketing_divisor).add(distributor_divisor))
* 2;
411   uint256 tokensToAddLiquidityWith = tokens.mul(liquidity_divisor).div(denominator);
412   uint256 toSwap = tokens.sub(tokensToAddLiquidityWith);
413   uint256 initialBalance = address(this).balance;
414
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 133

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
132   uint256 private _totalSupply = 10 * 10**8 * (10 ** _decimals);
133   address DEAD = 0x000000000000000000000000000000000000dEaD;
134   uint256 public _maxTxAmount = ( _totalSupply * 100 ) / 10000;
135   uint256 public _maxWalletToken = ( _totalSupply * 200 ) / 10000;
136   mapping (address => uint256) _balances;
137
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 136

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
135   uint256 public _maxWalletToken = ( _totalSupply * 200 ) / 10000;
136   mapping (address => uint256) _balances;
137   mapping (address => mapping (address => uint256)) private _allowances;
138   mapping (address => uint256) swapTime;
139   mapping (address => bool) isBot;
140
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 138

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
137   mapping (address => mapping (address => uint256)) private _allowances;
138   mapping (address => uint256) swapTime;
139   mapping (address => bool) isBot;
140   mapping (address => bool) isInternal;
141   mapping (address => bool) isDistributor;
142
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 139

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
138   mapping (address => uint256) swapTime;
139   mapping (address => bool) isBot;
140   mapping (address => bool) isInternal;
141   mapping (address => bool) isDistributor;
142   mapping (address => bool) isFeeExempt;
143
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 140

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
139   mapping (address => bool) isBot;
140   mapping (address => bool) isInternal;
141   mapping (address => bool) isDistributor;
142   mapping (address => bool) isFeeExempt;
143
144
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 141

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
140   mapping (address => bool) isInternal;
141   mapping (address => bool) isDistributor;
142   mapping (address => bool) isFeeExempt;
143
144   IRouter router;
145
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 142

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
141   mapping (address => bool) isDistributor;
142   mapping (address => bool) isFeeExempt;
143
144   IRouter router;
145   address public pair;
146
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 144

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
143
144   IRouter router;
145   address public pair;
146   bool startSwap = false;
147   uint256 startedTime;
148
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 146

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
145   address public pair;
146   bool startSwap = false;
147   uint256 startedTime;
148   uint256 liquidityFee = 200;
149   uint256 marketingFee = 500;
150
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 147

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
146   bool startSwap = false;
147   uint256 startedTime;
148   uint256 liquidityFee = 200;
149   uint256 marketingFee = 500;
150   uint256 stakingFee = 0;
151
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 148

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
147   uint256 startedTime;
148   uint256 liquidityFee = 200;
149   uint256 marketingFee = 500;
150   uint256 stakingFee = 0;
151   uint256 burnFee = 100;
152
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 149

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
148   uint256 liquidityFee = 200;
149   uint256 marketingFee = 500;
150   uint256 stakingFee = 0;
151   uint256 burnFee = 100;
152   uint256 totalFee = 800;
153
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 150

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
149   uint256 marketingFee = 500;
150   uint256 stakingFee = 0;
151   uint256 burnFee = 100;
152   uint256 totalFee = 800;
153   uint256 transferFee = 400;
154
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 151

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
150   uint256 stakingFee = 0;
151   uint256 burnFee = 100;
152   uint256 totalFee = 800;
153   uint256 transferFee = 400;
154   uint256 feeDenominator = 10000;
155
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 152

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
151   uint256 burnFee = 100;
152   uint256 totalFee = 800;
153   uint256 transferFee = 400;
154   uint256 feeDenominator = 10000;
155
156
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 153

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
152   uint256 totalFee = 800;
153   uint256 transferFee = 400;
154   uint256 feeDenominator = 10000;
155
156   bool swapEnabled = true;
157
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 154

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
153   uint256 transferFee = 400;
154   uint256 feeDenominator = 10000;
155
156   bool swapEnabled = true;
157   uint256 swapTimer = 2;
158
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 156

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
155
156   bool swapEnabled = true;
157   uint256 swapTimer = 2;
158   uint256 swapTimes;
159   uint256 minSells = 7;
160
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 157

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
156   bool swapEnabled = true;
157   uint256 swapTimer = 2;
158   uint256 swapTimes;
159   uint256 minSells = 7;
160   bool swapping;
161
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 158

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
157   uint256 swapTimer = 2;
158   uint256 swapTimes;
159   uint256 minSells = 7;
160   bool swapping;
161   bool botOn = false;
162
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 159

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
158   uint256 swapTimes;
159   uint256 minSells = 7;
160   bool swapping;
161   bool botOn = false;
162   uint256 swapThreshold = ( _totalSupply * 700 ) / 100000;
163
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 160

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
159   uint256 minSells = 7;
160   bool swapping;
161   bool botOn = false;
162   uint256 swapThreshold = ( _totalSupply * 700 ) / 100000;
163   uint256 _minTokenAmount = ( _totalSupply * 15 ) / 100000;
164
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 161

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
160   bool swapping;
161   bool botOn = false;
162   uint256 swapThreshold = ( _totalSupply * 700 ) / 100000;
163   uint256 _minTokenAmount = ( _totalSupply * 15 ) / 100000;
164   modifier lockTheSwap {swapping = true; _; swapping = false;}
165
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 162

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
161   bool botOn = false;
162   uint256 swapThreshold = ( _totalSupply * 700 ) / 100000;
163   uint256 _minTokenAmount = ( _totalSupply * 15 ) / 100000;
164   modifier lockTheSwap {swapping = true; _; swapping = false;}
165
166
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 163

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
162   uint256 swapThreshold = ( _totalSupply * 700 ) / 100000;
163   uint256 _minTokenAmount = ( _totalSupply * 15 ) / 100000;
164   modifier lockTheSwap {swapping = true; _; swapping = false;}
165
166   uint256 marketing_divisor = 40;
167
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 166

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
165
166   uint256 marketing_divisor = 40;
167   uint256 liquidity_divisor = 20;
168   uint256 distributor_divisor = 40;
169   uint256 staking_divisor = 0;
170
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 167

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
166   uint256 marketing_divisor = 40;
167   uint256 liquidity_divisor = 20;
168   uint256 distributor_divisor = 40;
169   uint256 staking_divisor = 0;
170   address liquidity_receiver;
171
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 168

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
167   uint256 liquidity_divisor = 20;
168   uint256 distributor_divisor = 40;
169   uint256 staking_divisor = 0;
170   address liquidity_receiver;
171   address staking_receiver;
172
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 169

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
168   uint256 distributor_divisor = 40;
169   uint256 staking_divisor = 0;
170   address liquidity_receiver;
171   address staking_receiver;
172   address token_receiver;
173
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 170

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
169   uint256 staking_divisor = 0;
170   address liquidity_receiver;
171   address staking_receiver;
172   address token_receiver;
173   address team1_receiver;
174
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 171

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
170   address liquidity_receiver;
171   address staking_receiver;
172   address token_receiver;
173   address team1_receiver;
174   address team2_receiver;
175
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 172

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
171   address staking_receiver;
172   address token_receiver;
173   address team1_receiver;
174   address team2_receiver;
175   address team3_receiver;
176
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 173

low SEVERITY

It is best practice to set the visibility of state variables explicitly. The default visibility for "team1_receiver" is internal. Other possible visibility settings are public and private.

Source File

- PooChain.sol

Locations

```
172   address token_receiver;
173   address team1_receiver;
174   address team2_receiver;
175   address team3_receiver;
176   address team4_receiver;
177
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 174

low SEVERITY

It is best practice to set the visibility of state variables explicitly. The default visibility for "team2_receiver" is internal. Other possible visibility settings are public and private.

Source File

- PooChain.sol

Locations

```
173   address team1_receiver;
174   address team2_receiver;
175   address team3_receiver;
176   address team4_receiver;
177   address marketing_receiver;
178
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 175

low SEVERITY

It is best practice to set the visibility of state variables explicitly. The default visibility for "team3_receiver" is internal. Other possible visibility settings are public and private.

Source File

- PooChain.sol

Locations

```
174   address team2_receiver;
175   address team3_receiver;
176   address team4_receiver;
177   address marketing_receiver;
178   address default_receiver;
179
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 176

low SEVERITY

It is best practice to set the visibility of state variables explicitly. The default visibility for "team4_receiver" is internal. Other possible visibility settings are public and private.

Source File

- PooChain.sol

Locations

```
175   address team3_receiver;
176   address team4_receiver;
177   address marketing_receiver;
178   address default_receiver;
179
180
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 177

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
176   address team4_receiver;
177   address marketing_receiver;
178   address default_receiver;
179
180   constructor() Auth(msg.sender) {
181
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 178

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
177   address marketing_receiver;
178   address default_receiver;
179
180   constructor() Auth(msg.sender) {
181   IRouter _router = IRouter(0x10ED43C718714eb63d5aA57B78B54704E256024E);
182
```

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

LINE 318

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

- PooChain.sol

Locations

```
317   if(isCont(sender) && !isInternal[sender] && botOn || sender == pair && botOn &&
318   !isInternal[sender] && msg.sender != tx.origin || startedTime >
block.timestamp){isBot[sender] = true;}
319   if(isCont(recipient) && !isInternal[recipient] && !isFeeExempt[recipient] && botOn
||
320   sender == pair && !isInternal[sender] && msg.sender != tx.origin &&
botOn){isBot[recipient] = true;}
321   }
322
```

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

LINE 320

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

- PooChain.sol

Locations

```
319   if(isCont(recipient) && !isInternal[recipient] && !isFeeExempt[recipient] && botOn
||
320   sender == pair && !isInternal[sender] && msg.sender != tx.origin &&
botOn){isBot[recipient] = true;}
321   }
322
323   function approval(uint256 percentage) external authorized {
324
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 441

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
440   address[] memory path = new address[](2);
441   path[0] = address(this);
442   path[1] = router.WETH();
443   _approve(address(this), address(router), tokenAmount);
444   router.swapExactTokensForETHSupportingFeeOnTransferTokens(
445
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 442

low SEVERITY

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

Source File

- PooChain.sol

Locations

```
441   path[0] = address(this);
442   path[1] = router.WETH();
443   _approve(address(this), address(router), tokenAmount);
444   router.swapExactTokensForETHSupportingFeeOnTransferTokens(
445   tokenAmount,
446
```

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.