



# CannaChain Smart Contract Audit Report

# 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
CannaChain	CNC	BSC

## Addresses

Contract address	0xD8fB6deb6f2045AE5B5Ed91642a35abf9502D70D
Contract deployer address	0x7E96d38F2DFE5FB6Afe9d8D7923CD66fD219D4C7

## Project Website

<a href="https://cannachain.cc/">https://cannachain.cc/</a>
---

## Codebase

<a href="https://bscscan.com/address/0xD8fB6deb6f2045AE5B5Ed91642a35abf9502D70D#code">https://bscscan.com/address/0xD8fB6deb6f2045AE5B5Ed91642a35abf9502D70D#code</a>
---

# SUMMARY

CannaChain is the world's first blockchain engine for cannabis, intended to solve the industry's problems of banking and payments, provenance & fundraising by using blockchain solutions. CannaChain aims to deliver secure payments, data on-chain, asset tokenization, and a complete, legal, and closed-loop cannabis ecosystem.

## Contract Summary

### Documentation Quality

CannaChain provides a very good documentation with standard of solidity base code.

- The technical description is provided clearly and structured and also don't have any high risk issue.

### Code Quality

The Overall quality of the basecode is standard.

- Standard solidity basecode and rules are already followed with CannaChain 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 286, 293, 301, 314, 316, 324, 325, 480, 487, 489, 490, 491, 548, 549, 555, 562, 563, 576, 602, 634, 647, 661, 662, 663, 664, 665, 666, 671, 692, 696, 697, 701, 704, 705, 712, 713, and 714.
- SWC-110 | It is recommended to use `revert()`, `assert()`, and `require()` in Solidity, and the new REVERT opcode in the EVM on lines 577, 577, 681, and 682.
- SWC-115 | `tx.origin` should not be used for authorization, use `msg.sender` instead on lines 634 and 635.
- SWC-120 | It is recommended to use external sources of randomness via oracles on lines 525, 634, 634, 635, and 636.

# CONCLUSION

We have audited the Goge Coin which has released on January 2023 to discover issues and identify potential security vulnerabilities in Goge Project. This process is used to find bugs, technical issues, and security loopholes that find some common issues in the code.

The security audit report produced satisfactory results with a low risk issue on the contract project.

The most common issue found in writing code on contracts that do not pose a big risk, writing on contracts is close to the standard of writing contracts in general. Some of the low issues that were found assert violation, authorization through tx.origin, and weak sources of randomness. We recommended Don't use any of those environment variables as sources of randomness and being aware that the use of these variables introduces a certain level of trust in miners, aware Using "tx.origin" as a security control can lead to authorization bypass vulnerabilities. Consider using "msg.sender" unless you really know what you are doing or going to do.

# 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
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
Assert Violation	SWC-110	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 Caller	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
Shadowing State Variable	SWC-119	State variables should not be shadowed.	PASS
Weak Sources of Randomness	SWC-120	Random values should never be generated from Chain Attributes or be predictable.	ISSUE FOUND
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

# SMART CONTRACT ANALYSIS

Started	Sat Jan 14 2023 04:23:38 GMT+0000 (Coordinated Universal Time)
Finished	Mon Jan 15 2023 05:13:30 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	CannaChain.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

<b>SWC-101</b>	ARITHMETIC OPERATION "*" DISCOVERED	<b>low</b>	acknowledged
<b>SWC-101</b>	ARITHMETIC OPERATION "*" DISCOVERED	<b>low</b>	acknowledged
<b>SWC-101</b>	ARITHMETIC OPERATION "/" DISCOVERED	<b>low</b>	acknowledged
<b>SWC-101</b>	ARITHMETIC OPERATION "*" DISCOVERED	<b>low</b>	acknowledged
<b>SWC-101</b>	ARITHMETIC OPERATION "-=" DISCOVERED	<b>low</b>	acknowledged
<b>SWC-101</b>	ARITHMETIC OPERATION "-=" DISCOVERED	<b>low</b>	acknowledged
<b>SWC-101</b>	ARITHMETIC OPERATION "/" DISCOVERED	<b>low</b>	acknowledged
<b>SWC-101</b>	ARITHMETIC OPERATION "/" DISCOVERED	<b>low</b>	acknowledged
<b>SWC-101</b>	ARITHMETIC OPERATION "/" DISCOVERED	<b>low</b>	acknowledged
<b>SWC-115</b>	USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.	<b>low</b>	acknowledged
<b>SWC-115</b>	USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.	<b>low</b>	acknowledged
<b>SWC-110</b>	OUT OF BOUNDS ARRAY ACCESS	<b>low</b>	acknowledged
<b>SWC-110</b>	OUT OF BOUNDS ARRAY ACCESS	<b>low</b>	acknowledged
<b>SWC-110</b>	OUT OF BOUNDS ARRAY ACCESS	<b>low</b>	acknowledged
<b>SWC-120</b>	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	<b>low</b>	acknowledged
<b>SWC-120</b>	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	<b>low</b>	acknowledged
<b>SWC-120</b>	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	<b>low</b>	acknowledged
<b>SWC-120</b>	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	<b>low</b>	acknowledged

# SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 286

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
285     unchecked {  
286         _approve(sender, _msgSender(), currentAllowance - amount);  
287     }  
288  
289     return true;  
290
```

# SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 293

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
292     function increaseAllowance(address spender, uint256 addedValue) public virtual
returns (bool) {
293     _approve(_msgSender(), spender, _allowances[_msgSender()][spender] + addedValue);
294     return true;
295 }
296
297
```

# SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 301

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
300     unchecked {  
301         _approve(_msgSender(), spender, currentAllowance - subtractedValue);  
302     }  
303  
304     return true;  
305
```

## SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 314

### low SEVERITY

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

### Source File

- CannaChain.sol

### Locations

```
313     unchecked {  
314         _balances[sender] = senderBalance - amount;  
315     }  
316     _balances[recipient] += amount;  
317  
318
```

## SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 316

### low SEVERITY

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

### Source File

- CannaChain.sol

### Locations

```
315     }  
316     _balances[recipient] += amount;  
317  
318     emit Transfer(sender, recipient, amount);  
319     }  
320
```

# SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 324

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
323
324   _totalSupply += amount;
325   _balances[account] += amount;
326   emit Transfer(address(0), account, amount);
327   }
328
```



# SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 325

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
324  _totalSupply += amount;  
325  _balances[account] += amount;  
326  emit Transfer(address(0), account, amount);  
327  }  
328  
329
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 480

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
479
480  uint256 totalSupply = 420_000 * 1e18;
481
482  sellMarketingFee = 300;
483  sellLiquidityFee = 100;
484
```

# SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 487

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
486     sellCannaFundFee = 200;  
487     sellTotalFees = sellMarketingFee + sellLiquidityFee + sellTeamFee + sellBuyBackFee  
+     sellCannaFundFee;  
488  
489     maxTxnAmount = totalSupply * 1 / 100;  
490     maxWallet = totalSupply * 1 / 100;  
491
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 489

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
488
489     maxTxnAmount = totalSupply * 1 / 100;
490     maxWallet = totalSupply * 1 / 100;
491     swapTokensAtAmount = totalSupply * 25 / 100000;
492
493
```

## SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 490

### low SEVERITY

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

### Source File

- CannaChain.sol

### Locations

```
489     maxTxnAmount = totalSupply * 1 / 100;  
490     maxWallet = totalSupply * 1 / 100;  
491     swapTokensAtAmount = totalSupply * 25 / 100000;  
492  
493     //@dev update these!  
494
```

# SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 491

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
490     maxWallet = totalSupply * 1 / 100;  
491     swapTokensAtAmount = totalSupply * 25 / 100000;  
492  
493     //@dev update these!  
494     marketingAddress = address(0x7c537E135A5B5E3d0E61C30118683f2F800443f3);  
495
```

# SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 548

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
547     require(limitsInEffect, "Limits already lifted");
548     require(newNum >= (totalSupply() * 1 / 1000)/1e18, "Cannot set max txn lower than
0.1%");
549     maxTxnAmount = newNum * (10**18);
550     emit UpdatedMaxTxnAmount(maxTxnAmount);
551 }
552
```

## SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 549

### low SEVERITY

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

### Source File

- CannaChain.sol

### Locations

```
548   require(newNum >= (totalSupply() * 1 / 1000)/1e18, "Cannot set max txn lower than
549   0.1%");
549   maxTxnAmount = newNum * (10**18);
550   emit UpdatedMaxTxnAmount(maxTxnAmount);
551   }
552
553
```



## SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 549

### low SEVERITY

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

### Source File

- CannaChain.sol

### Locations

```
548     require(newNum >= (totalSupply() * 1 / 1000)/1e18, "Cannot set max txn lower than
549         0.1%");
549     maxTxnAmount = newNum * (10**18);
550     emit UpdatedMaxTxnAmount(maxTxnAmount);
551 }
552
553
```

## SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 555

### low SEVERITY

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

### Source File

- CannaChain.sol

### Locations

```
554     require(limitsInEffect, "Limits already lifted");
555     require(newNum >= (totalSupply() * 1 / 100)/1e18, "Cannot set max txn lower than
1%");
556     maxWallet = newNum * (10**18);
557     emit UpdatedMaxWallet(maxTxnAmount);
558 }
559
```

## SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 556

### low SEVERITY

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

### Source File

- CannaChain.sol

### Locations

```
555   require(newNum >= (totalSupply() * 1 / 100)/1e18, "Cannot set max txn lower than
1%");
556   maxWallet = newNum * (10**18);
557   emit UpdatedMaxWallet(maxTxnAmount);
558   }
559
560
```

## SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 562

### low SEVERITY

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

### Source File

- CannaChain.sol

### Locations

```
561     function updateSwapTokensAtAmount(uint256 newAmount) external onlyOwner {
562         require(newAmount >= totalSupply() * 1 / 100000, "Swap amount cannot be lower than
0.001% total supply.");
563         require(newAmount <= totalSupply() * 1 / 1000, "Swap amount cannot be higher than
0.1% total supply.");
564         swapTokensAtAmount = newAmount;
565         emit SwapTokensAtAmountUpdated(newAmount);
566     }
```

# SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 563

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
562     require(newAmount >= totalSupply() * 1 / 100000, "Swap amount cannot be lower than
0.001% total supply.");
563     require(newAmount <= totalSupply() * 1 / 1000, "Swap amount cannot be higher than
0.1% total supply.");
564     swapTokensAtAmount = newAmount;
565     emit SwapTokensAtAmountUpdated(newAmount);
566 }
567
```

# SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 576

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
575     require(wallets.length < 600, "Can only airdrop 600 wallets per txn due to gas
limits");
576     for(uint256 i = 0; i < wallets.length; i++){
577         super._transfer(msg.sender, wallets[i], amountsInTokens[i]);
578     }
579 }
580
```

# SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 602

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
601     sellCannaFundFee = _cannaFundFee;
602     sellTotalFees = sellMarketingFee + sellLiquidityFee + sellTeamFee + sellBuyBackFee
+ sellCannaFundFee;
603     require(sellTotalFees <= 2500, "Must keep sell fees at 25% or less");
604     emit UpdatedSellFee(sellTotalFees);
605 }
606
```

# SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 634

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
633   if (to != address(dexRouter) && to != address(lpPair)){  
634       require(_holderLastTransferBlock[tx.origin] + 5 < block.number &&  
_holderLastTransferBlock[to] + 5 < block.number, "_transfer:: Transfer Delay enabled.  
Try again later.");  
635       _holderLastTransferBlock[tx.origin] = block.number;  
636       _holderLastTransferBlock[to] = block.number;  
637   }  
638
```



# SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 643

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
642   require(amount <= maxTxnAmount, "Max txn exceeded.");
643   require(amount + balanceOf(to) <= maxWallet, "Max Wallet Exceeded");
644   } else if (automatedMarketMakerPairs[to] && !_isExcludedMaxTransactionAmount[from])
{
645   require(amount <= maxTxnAmount, "Max txn exceeded.");
646   } else if (!_isExcludedMaxTransactionAmount[to]){
647
```

## SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 647

### low SEVERITY

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

### Source File

- CannaChain.sol

### Locations

```
646     } else if (!_isExcludedMaxTransactionAmount[to]){  
647         require(amount + balanceOf(to) <= maxWallet, "Max Wallet Exceeded");  
648     }  
649 }  
650  
651
```

# SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 661

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
660   if (automatedMarketMakerPairs[to] && sellTotalFees > 0){  
661     fees = amount * sellTotalFees / FEE_DIVISOR;  
662     tokensForLiquidity += fees * sellLiquidityFee / sellTotalFees;  
663     tokensForMarketing += fees * sellMarketingFee / sellTotalFees;  
664     tokensForTeam += fees * sellTeamFee / sellTotalFees;  
665
```

# SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 662

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
661 fees = amount * sellTotalFees / FEE_DIVISOR;
662 tokensForLiquidity += fees * sellLiquidityFee / sellTotalFees;
663 tokensForMarketing += fees * sellMarketingFee / sellTotalFees;
664 tokensForTeam += fees * sellTeamFee / sellTotalFees;
665 tokensForBuyBack += fees * sellBuyBackFee / sellTotalFees;
666
```

# SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 663

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
662 tokensForLiquidity += fees * sellLiquidityFee / sellTotalFees;  
663 tokensForMarketing += fees * sellMarketingFee / sellTotalFees;  
664 tokensForTeam += fees * sellTeamFee / sellTotalFees;  
665 tokensForBuyBack += fees * sellBuyBackFee / sellTotalFees;  
666 tokensForCannaFund += fees * sellCannaFundFee / sellTotalFees;  
667
```

# SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 664

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
663     tokensForMarketing += fees * sellMarketingFee / sellTotalFees;  
664     tokensForTeam += fees * sellTeamFee / sellTotalFees;  
665     tokensForBuyBack += fees * sellBuyBackFee / sellTotalFees;  
666     tokensForCannaFund += fees * sellCannaFundFee / sellTotalFees;  
667 }  
668
```

# SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 665

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
664 tokensForTeam += fees * sellTeamFee / sellTotalFees;  
665 tokensForBuyBack += fees * sellBuyBackFee / sellTotalFees;  
666 tokensForCannaFund += fees * sellCannaFundFee / sellTotalFees;  
667 }  
668  
669
```

# SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 666

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
665     tokensForBuyBack += fees * sellBuyBackFee / sellTotalFees;  
666     tokensForCannaFund += fees * sellCannaFundFee / sellTotalFees;  
667 }  
668  
669     if(fees > 0){  
670
```



# SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 671

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
670  super._transfer(from, address(this), fees);
671  amount -= fees;
672  }
673
674  super._transfer(from, to, amount);
675
```

## SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 692

### low SEVERITY

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

### Source File

- CannaChain.sol

### Locations

```
691  uint256 contractBalance = balanceOf(address(this));
692  uint256 totalTokensToSwap = tokensForTeam + tokensForMarketing + tokensForLiquidity
+ tokensForBuyBack + tokensForCannaFund;
693
694  if(contractBalance == 0 || totalTokensToSwap == 0) {return;}
695
696
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 696

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
695
696   if(contractBalance > swapTokensAtAmount * 60){
697       contractBalance = swapTokensAtAmount * 60;
698   }
699
700
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 697

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
696     if(contractBalance > swapTokensAtAmount * 60){  
697         contractBalance = swapTokensAtAmount * 60;  
698     }  
699  
700     if(tokensForLiquidity > 0){  
701
```

# SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 701

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
700     if(tokensForLiquidity > 0){  
701         uint256 liquidityTokens = contractBalance * tokensForLiquidity / totalTokensToSwap;  
702         super._transfer(address(this), lpPair, liquidityTokens);  
703         try ILpPair(lpPair).sync(){} catch {}  
704         contractBalance -= liquidityTokens;  
705     }
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 701

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
700     if(tokensForLiquidity > 0){  
701         uint256 liquidityTokens = contractBalance * tokensForLiquidity / totalTokensToSwap;  
702         super._transfer(address(this), lpPair, liquidityTokens);  
703         try ILpPair(lpPair).sync(){} catch {}  
704         contractBalance -= liquidityTokens;  
705     }
```

## SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 704

### low SEVERITY

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

### Source File

- CannaChain.sol

### Locations

```
703     try ILpPair(lpPair).sync(){} catch {}  
704     contractBalance -= liquidityTokens;  
705     totalTokensToSwap -= tokensForLiquidity;  
706 }  
707  
708
```

# SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 705

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
704 contractBalance -= liquidityTokens;  
705 totalTokensToSwap -= tokensForLiquidity;  
706 }  
707  
708 swapTokensForEth(contractBalance);  
709
```



# SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 712

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
711
712  uint256 ethForMarketing = ethBalance * tokensForMarketing / totalTokensToSwap;
713  uint256 ethForBuyBack = ethBalance * tokensForBuyBack / totalTokensToSwap;
714  uint256 ethForCannaFund = ethBalance * tokensForCannaFund / totalTokensToSwap;
715
716
```

# SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 713

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
712 uint256 ethForMarketing = ethBalance * tokensForMarketing / totalTokensToSwap;  
713 uint256 ethForBuyBack = ethBalance * tokensForBuyBack / totalTokensToSwap;  
714 uint256 ethForCannaFund = ethBalance * tokensForCannaFund / totalTokensToSwap;  
715  
716 tokensForTeam = 0;  
717
```

# SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 714

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
713     uint256 ethForBuyBack = ethBalance * tokensForBuyBack / totalTokensToSwap;  
714     uint256 ethForCannaFund = ethBalance * tokensForCannaFund / totalTokensToSwap;  
715  
716     tokensForTeam = 0;  
717     tokensForMarketing = 0;  
718
```

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

LINE 634

## 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

- CannaChain.sol

## Locations

```
633   if (to != address(dexRouter) && to != address(lpPair)){  
634       require(_holderLastTransferBlock[tx.origin] + 5 < block.number &&  
_holderLastTransferBlock[to] + 5 < block.number, "_transfer:: Transfer Delay enabled.  
Try again later.");  
635       _holderLastTransferBlock[tx.origin] = block.number;  
636       _holderLastTransferBlock[to] = block.number;  
637   }  
638
```

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

LINE 635

## low SEVERITY

Using "tx.origin" as a security control can lead to authorization bypass vulnerabilities. Consider using "msg.sender" unless you really know what you are doing.

## Source File

- CannaChain.sol

## Locations

```
634   require(_holderLastTransferBlock[tx.origin] + 5 < block.number &&
_holderLastTransferBlock[to] + 5 < block.number, "_transfer:: Transfer Delay enabled.
Try again later.");
635   _holderLastTransferBlock[tx.origin] = block.number;
636   _holderLastTransferBlock[to] = block.number;
637   }
638   }
639
```

## 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

- CannaChain.sol

### Locations

```
576   for(uint256 i = 0; i < wallets.length; i++){  
577     super._transfer(msg.sender, wallets[i], amountsInTokens[i]);  
578   }  
579 }  
580  
581
```

## SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 681

### low SEVERITY

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

### Source File

- CannaChain.sol

### Locations

```
680     address[] memory path = new address[](2);
681     path[0] = address(this);
682     path[1] = dexRouter.WETH();
683
684     // make the swap
685
```

# SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 682

## low SEVERITY

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

## Source File

- CannaChain.sol

## Locations

```
681     path[0] = address(this);
682     path[1] = dexRouter.WETH();
683
684     // make the swap
685     dexRouter.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount, 0, path,
address(this), block.timestamp);
686
```



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

LINE 525

### low SEVERITY

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

### Source File

- CannaChain.sol

### Locations

```
524 swapEnabled = true;
525 tradingActiveBlock = block.number;
526 emit EnabledTrading();
527 }
```

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

LINE 634

### low SEVERITY

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

### Source File

- CannaChain.sol

### Locations

```
633   if (to != address(dexRouter) && to != address(lpPair)){  
634       require(_holderLastTransferBlock[tx.origin] + 5 < block.number &&  
_holderLastTransferBlock[to] + 5 < block.number, "_transfer:: Transfer Delay enabled.  
Try again later.");  
635       _holderLastTransferBlock[tx.origin] = block.number;  
636       _holderLastTransferBlock[to] = block.number;  
637   }
```

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

LINE 635

### low SEVERITY

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

### Source File

- CannaChain.sol

### Locations

```
634   require(_holderLastTransferBlock[tx.origin] + 5 < block.number &&
_holderLastTransferBlock[to] + 5 < block.number, "_transfer:: Transfer Delay enabled.
Try again later.");
635   _holderLastTransferBlock[tx.origin] = block.number;
636   _holderLastTransferBlock[to] = block.number;
637   }
638   }
```

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

LINE 636

### low SEVERITY

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

### Source File

- CannaChain.sol

### Locations

```
635 _holderLastTransferBlock[tx.origin] = block.number;  
636 _holderLastTransferBlock[to] = block.number;  
637 }  
638 }
```

# 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.