



SAFUAUDIT

SMART CONTRACT AUDITING

MYLOCALPAX

SMART CONTRACT AUDIT



March 17, 2022

INTRODUCTION

Client	MyLocalPax(MLP)
Language	Solidity
Contract address	0x75eA8C8d5e79F9Aad474C355CC38E47E650B2424
Decimals	9
Supply	1,000,000,000,000
Platform	Binance Smart Chain
Compiler	v0.6.12+commit.27d51765
Optimization	Yes, with 200 runs
Website	https://mylocalpax.com/
Telegram	https://t.me/mylocalpax
Twitter	https://twitter.com/mylocalpax

Description

MyLocalPax is a p2p crypto exchange marketplace where people can trade crypto directly from over 3 million users worldwide. If you're buying, send payment directly to the seller; if you're selling, send Bitcoin Cash to our blind escrow. Once payment is received by the seller, the BCH is released to the buyer.

TABLE OF CONTENTS

01 INTRODUCTION

Introduction	02
Approach	04
Risk classification	05

02 ABSTRACT

Abstract	06
----------	----

03 VULNERABILITIES TEST

Vulnerabilities Test	07
----------------------	----

04 MANUAL ANALYSIS

Manual analysis	09
Contract Inspection	10
Inheritance Tree	13
Important Snippets	14
Good Practices	15

05 WEBSITE

Website Audit	16
---------------	----

06 CONCLUSIONS

Disclaimer	17
Audit Results	18
SafuScore	19
Summary	20

Approach



Audit Details

Our comprehensive audit report provides a full overview of the audited system's architecture, smart contract codebase, and details on any vulnerabilities found within the system.



Audit Goals

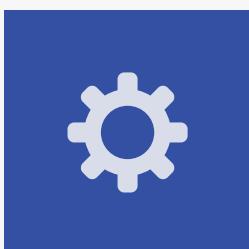
The audit goal is to ensure that the project is built to protect investors and users, preventing potentially catastrophic vulnerabilities after launch, that lead to scams and rugpulls.



Code Quality

Our analysis includes both automatic tests and manual code analysis for the following aspects:

- Exploits
 - Back-doors
 - Vulnerability
 - Accuracy
 - Readability
-



Tools

- Remix IDE
- MythX, Mytrhl
- SWC Registry
- Open Zeppelin Code Analyzer
- Solidity Code Complier

RISK CLASSIFICATION

CRITICAL

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment

MEDIUM

Issues on this level could potentially bring problems and should eventually be fixed.

MINOR

Issues on this level are minor details and warning that can remain unfixed but would be better fixed at some point in the future

INFORMATIONAL

Information level is to offer suggestions for improvement of efficacy or security for features with a risk free factor.

ABSTRACT

Fees	Ownership	Mint function
Buy Fees: 7% Sell Fees: 16% <small>*at audit time</small>	Owned	No mint function
Owner can't set fees	Max Tx amount	Pause
Owner can change fees up to 100%	Owner can set max Tx amount	Owner can't pause trading

Vulnerabilities Test

SWC ID	Description	
SWC-100	Function Default Visibility	Passed
SWC-101	Integer Overflow and Underflow	Passed
SWC-102	Outdated Compiler Version	Passed
SWC-103	FloatingPragma	Minor
SWC-104	Unchecked Call Return Value	Passed
SWC-105	Unprotected Ether Withdrawal	Passed
SWC-106	Unprotected SELF-DESTRUCT Instruction	Passed
SWC-107	Re-entrancy	Passed
SWC-108	State Variable Default Visibility	Minor
SWC-109	Uninitialized Storage Pointer	Passed
SWC-110	Assert Violation	Passed
SWC-111	Use of Deprecated Solidity Functions	Passed
SWC-112	Delegate Call to Untrusted Callee	Passed
SWC-113	DoS with Failed Call	Passed
SWC-114	Transaction Order Dependence	Passed
SWC-115	Authorization through tx.origin	Minor

SWC-116	Block values as a proxy for time	Passed
SWC-117	Signature Malleability	Passed
SWC-118	Incorrect Constructor Name	Passed
SWC-119	Shadowing State Variables	Passed
SWC-120	Weak Sources of Randomness from Chain Attributes	Passed
SWC-121	Missing Protection against Signature Replay Attacks	Passed
SWC-122	Lack of Proper Signature Verification	Passed
SWC-123	Requirement Violation	Passed
SWC-124	Write to Arbitrary Storage Location	Passed
SWC-125	Incorrect Inheritance Order	Passed
SWC-126	Insufficient Gas Griefing	Passed
SWC-127	Arbitrary Jump with Function Type Variable	Passed
SWC-128	DoS With Block Gas Limit	Passed
SWC-129	Typographical Error	Passed
SWC-130	Right-To-Left-Override control character (U+202E)	Passed
SWC-131	Presence of unused variables	Passed
SWC-132	Unexpected Ether balance	Passed
SWC-133	Hash Collisions With Multiple Variable Length Arguments	Passed
SWC-134	Message call with the hardcoded gas amount	Passed
SWC-135	Code With No Effects (Irrelevant/Dead Code)	Passed
SWC-136	Unencrypted Private Data On-Chain	Passed

MANUAL ANALYSIS

The contract is verified to check if functions do and work as they should and malicious code is not inserted.

	Tested	Result
Transfer	Yes	Passed
Total Supply	Yes	Passed
Buy Back	Yes	N/A
Burn	Yes	N/A
Mint	Yes	N/A
Rebase	Yes	N/A
Pause	Yes	N/A
Blacklist	Yes	N/A
Lock	Yes	N/A
Max Transaction	Yes	Passed
Transfer Ownership	Yes	Passed
Renounce Ownership	Yes	Passed

MANUAL AUDIT

CONTRACT INSPECTION



Context Implementation		
L _msgSender Internal	🔒	
L _msgData Internal	🔒	
IERC20 Interface		
L totalSupply External	❗	NO
L balanceOf External	❗	NO
L transfer External	❗	🔴 NO
L allowance External	❗	NO
L approve External	❗	🔴 NO
L transferFrom External	❗	🔴 NO
SafeMath Library		
L add Internal	🔒	
L sub Internal	🔒	
L sub Internal	🔒	
L mul Internal	🔒	
L div Internal	🔒	
L div Internal	🔒	
L mod Internal	🔒	
L mod Internal	🔒	
Address Library		
L isContract Internal	🔒	
L sendValue Internal	🔒	🔴
L functionCall Internal	🔒	🔴
L functionCall Internal	🔒	🔴
L functionCallWithValue Internal	🔒	🔴
L functionCallWithValue Internal	🔒	🔴
L _functionCallWithValue Private	🔒	🔴
Ownable Implementation Context		
L <Constructor> Internal	🔒	🔴
L owner Public	❗	NO
L renounceOwnership Public	❗	🔴 onlyOwner
L transferOwnership Public	❗	🔴 onlyOwner

```
| **IUniswapV2Factory** | Interface | ||| |
| L | createPair | External ! | ○ | NO! |
|||||
| **IUniswapV2Pair** | Interface | |||
| L | sync | External ! | ○ | NO! |
|||||
| **IUniswapV2Router01** | Interface | |||
| L | factory | External ! | | NO! |
| L | WETH | External ! | | NO! |
| L | addLiquidity | External ! | ○ | NO! |
| L | addLiquidityETH | External ! | ⚡ | NO! |
|||||
| **IUniswapV2Router02** | Interface | IUniswapV2Router01 ||
| L | removeLiquidityETHSupportingFeeOnTransferTokens | External ! | ○ | NO! |
| L | swapExactTokensForETHSupportingFeeOnTransferTokens | External ! | ○ | NO! |
| L | swapExactTokensForTokensSupportingFeeOnTransferTokens | External ! | ○ | NO! |
| L | swapExactETHForTokensSupportingFeeOnTransferTokens | External ! | ⚡ | NO! |
|||||
| **ReentrancyGuard** | Implementation | ||
| L | <Constructor> | Public ! | ○ | NO! |
|||||
| **TransferHelper** | Library | ||
| L | safeApprove | Internal 🔒 | ○ | ||
| L | safeTransfer | Internal 🔒 | ○ | ||
| L | safeTransferFrom | Internal 🔒 | ○ | ||
| L | safeTransferETH | Internal 🔒 | ○ | ||
|||||
| **Wallet** | Implementation | ||
| L | <Receive Ether> | External ! | ⚡ | NO! |
|||||
| **MyLocalPax** | Implementation | Context, IERC20, Ownable, ReentrancyGuard ||
| L | <Constructor> | Public ! | ○ | NO! |
| L | name | Public ! | | NO! |
| L | symbol | Public ! | | NO! |
| L | decimals | Public ! | | NO! |
| L | totalSupply | Public ! | | NO! |
| L | balanceOf | Public ! | | NO! |
| L | transfer | Public ! | ○ | NO! |
```

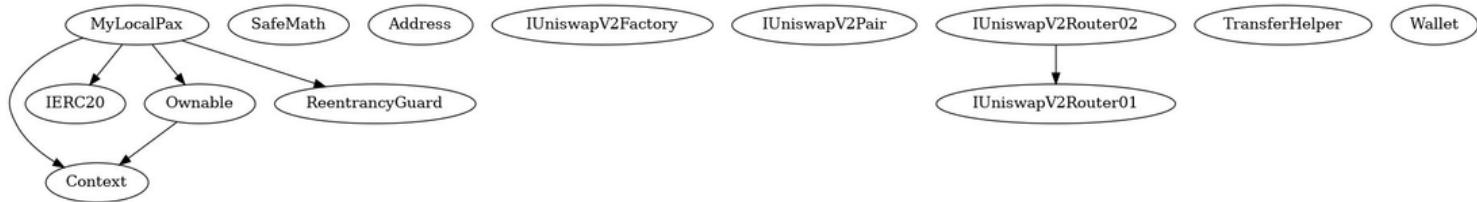
```

| L | allowance | Public ! | | NO! |
| L | approve | Public ! |  | NO! |
| L | transferFrom | Public ! |  | NO! |
| L | increaseAllowance | Public ! |  | NO! |
| L | decreaseAllowance | Public ! |  | NO! |
| L | isExcluded | Public ! | | NO! |
| L | reflectionFromToken | Public ! | | NO! |
| L | tokenFromReflection | Public ! | | NO! |
| L | excludeAccount | External ! |  | onlyOwner |
| L | includeAccount | External ! |  | onlyOwner |
| L | _approve | Private  |  | |
| L | _transfer | Private  |  | |
| L | collectFee | Private  |  | |
| L | calculateReward | Public ! | | NO! |
| L | claimReward | Public ! |  | isHuman nonReentrant lockTheSwap |
| L | topUpClaimCycleAfterTransfer | Private  |  | |
| L | calculateTopUpClaim | Public ! | | NO! |
| L | _getReflectionRate | Private  | | |
| L | swap | Private  |  | lockTheSwap |
| L | swapAndLiquify | Private  |  | |
| L | swapTokensForCake | Private  |  | |
| L | swapTokensForEth | Private  |  | |
| L | swapEthForTokens | Private  |  | |
| L | addLiquidity | Private  |  | |
| L | setPairRouterCake | External ! |  | onlyOwner |
| L | setTaxless | External ! |  | onlyOwner |
| L | setSwapEnabled | External ! |  | onlyOwner |
| L | setFeeActive | External ! |  | onlyOwner |
| L | setTaxFee | External ! |  | onlyOwner |
| L | setEscrowFee | External ! |  | onlyOwner |
| L | setdevelopmentFee | External ! |  | onlyOwner |
| L | setLiquidityFee | External ! |  | onlyOwner |
| L | setdevelopmentWallet | External ! |  | onlyOwner |
| L | setEscrowWallet | External ! |  | onlyOwner |
| L | setMaxTxAmount | External ! |  | onlyOwner |
| L | setMinTokensBeforeSwap | External ! |  | onlyOwner |
| L | setRewardCycleInterval | External ! |  | onlyOwner |
| L | <Receive Ether> | External ! |  | NO! |

```

Symbol	Meaning
	Function can modify state
	Function is payable
	Private function
	Internal function
NO!	Function has no modifier

INHERITANCE TREE



Inheritance is a feature of the object-oriented programming language. It is a way of extending the functionality of a program, used to separate the code, reduces the dependency, and increases the re-usability of the existing code. Solidity supports inheritance between smart contracts, where multiple contracts can be inherited into a single contract.

Important Snippets



Owner can set max Tx amount

```
function setMaxTxAmount(uint256 amount) external onlyOwner {  
    maxTxAmount = amount;  
}
```

Owner can enable/disable taxes on wallets

```
function setTaxless(address account, bool value) external onlyOwner {  
    isTaxless[account] = value;  
}
```

Owner can disable and enable swap

```
function setSwapEnabled(bool enabled) external onlyOwner {  
    swapEnabled = enabled;  
    SwapUpdated(enabled);  
}
```

Owner can set fees up to 100%

```
function setTaxFee(uint256 buy, uint256 sell, uint256 peer) external onlyOwner {  
    _taxFee[0] = buy;  
    _taxFee[1] = sell;  
    _taxFee[2] = peer;  
}  
  
function setEscrowFee(uint256 buy, uint256 sell, uint256 peer) external onlyOwner {  
    _EscrowFee[0] = buy;  
    _EscrowFee[1] = sell;  
    _EscrowFee[2] = peer;  
}  
  
function setdevelopmentFee(uint256 buy, uint256 sell, uint256 peer) external onlyOwner {  
    _developmentFee[0] = buy;  
    _developmentFee[1] = sell;  
    _developmentFee[2] = peer;  
}  
  
function setLiquidityFee(uint256 buy, uint256 sell, uint256 peer) external onlyOwner {  
    _liqFee[0] = buy;  
    _liqFee[1] = sell;  
    _liqFee[2] = peer;  
}
```

GOOD PRACTICES ✓

- The owner cannot mint new tokens after deployment
- The smart contract utilizes "SafeMath" to prevent overflows

```
library SafeMath {  
    function tryAdd(uint256 a, uint256 b) internal pure returns (bool, uint256) {  
        unchecked {  
            uint256 c = a + b;  
            if (c < a) return (false, 0);  
            return (true, c);  
        }  
    }  
  
    function trySub(uint256 a, uint256 b) internal pure returns (bool, uint256) {  
        unchecked {  
            if (b > a) return (false, 0);  
            return (true, a - b);  
        }  
    }  
  
    function tryMul(uint256 a, uint256 b) internal pure returns (bool, uint256) {  
        unchecked {  
            // Gas optimization: this is cheaper than requiring 'a' not being zero, but  
            // benefit is lost if 'b' is also tested.  
            // See: https://github.com/OpenZeppelin/openzeppelin-contracts/pull/522  
            if (a == 0) return (true, 0);  
            uint256 c = a * b;  
            if (c / a != b) return (false, 0);  
            return (true, c);  
        }  
    }  
  
    function tryDiv(uint256 a, uint256 b) internal pure returns (bool, uint256) {  
        unchecked {  
            if (b == 0) return (false, 0);  
            return (true, a / b);  
        }  
    }  
  
    function tryMod(uint256 a, uint256 b) internal pure returns (bool, uint256) {  
        unchecked {  
            if (b == 0) return (false, 0);  
            return (true, a % b);  
        }  
    }  
}
```

WEBSITE



Website	https://mylocalpax.com/
Domain Registry	http://www.ownregistrar.com
Domain Expiry Date	2023-03-15
Response Code	200
SSL Checker and HTTPS Test	Passed
Deprecated HTML tags	Informative
Robots.txt	Informative
Sitemap Test	Informative
SEO Friendly URL	Passed
Responsive Test	Passed
JS Error Test	Passed
Console Errors Test	Minor
Site Loading Speed Test	1.74 seconds - Passed
HTTP2 Test	Passed
Safe Browsing Test	Passed

DISCLAIMER

SafuAudit.com is not a financial institution and the information provided on this website does not constitute investment advice, financial advice, trading advice or any other sort of advice. You should not treat any of the website's content as such. Investing in crypto assets carries a high level of risk and does not hold guarantees for not sustaining financial loss due to their volatility.

Accuracy of Information

SafuAudit will strive to ensure accuracy of information listed on this website although it will not hold any responsibility for any missing or wrong information. SafuAudit provides all information as is. You understand that you are using any and all information available here at your own risk. Any use or reliance on our content and services is solely at your own risk and discretion.

The purpose of the audit is to analyse the on-chain smart contract source code, and to provide basic overview of the project.

While we have used all the information available to us for this straightforward investigation, you should not rely on this report only – we recommend proceeding with several independent audits. Be aware that smart contracts deployed on a blockchain aren't secured enough against external vulnerability, or a hack. Be aware that active smart contract owner privileges constitute an elevated impact to smart contract's safety and security. Therefore, SafuAudit does not guarantee the explicit security of the audited smart contract. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

AUDIT RESULTS

CRITICAL

No critical severity issues have been found.

MEDIUM

- Owner can set fees up to 100%
- Owner can set maxTx amount, if it is set to 0, transfers are blocked

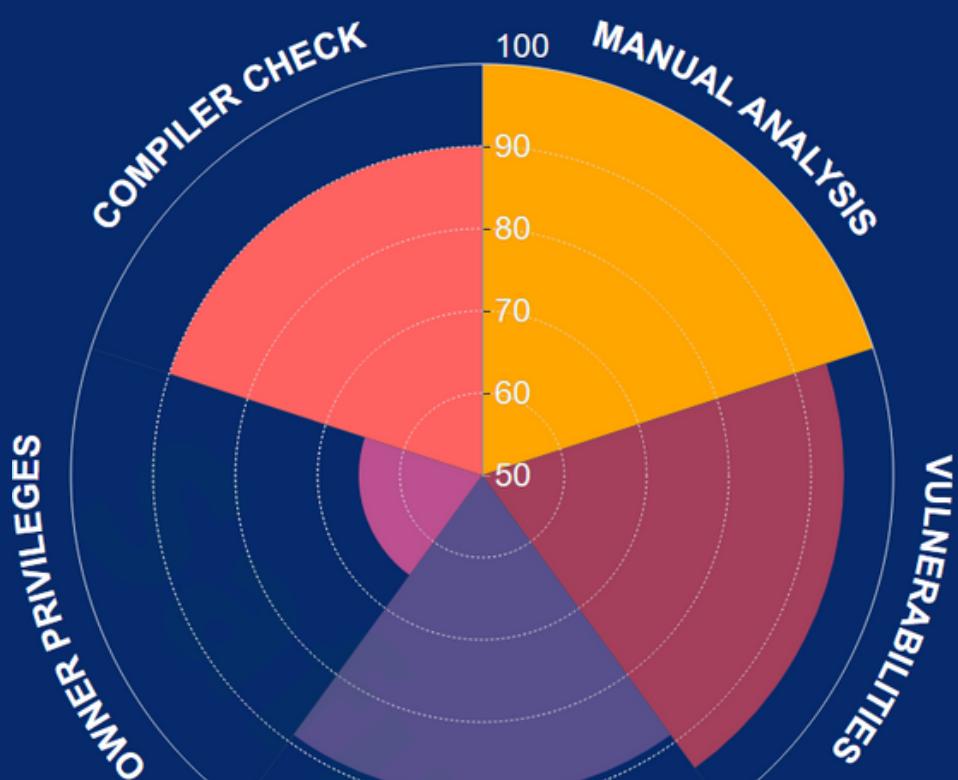
MINOR

- Using "tx.origin" as a security control can lead to authorization bypass vulnerabilities. Consider using "msg.sender" instead.
- A floating pragma is set. It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds.
- State variable visibility is not set. It is best practice to set the visibility of state variables explicitly. The default visibility for "isTaxless" is internal.

INFORMATIONAL

The standard audit model does not offer suggestions and consulting for improvements of efficacy.

SAFUSCORE



Manual Analysis



Vulnerabilities



Contract Readability



Owner Privileges

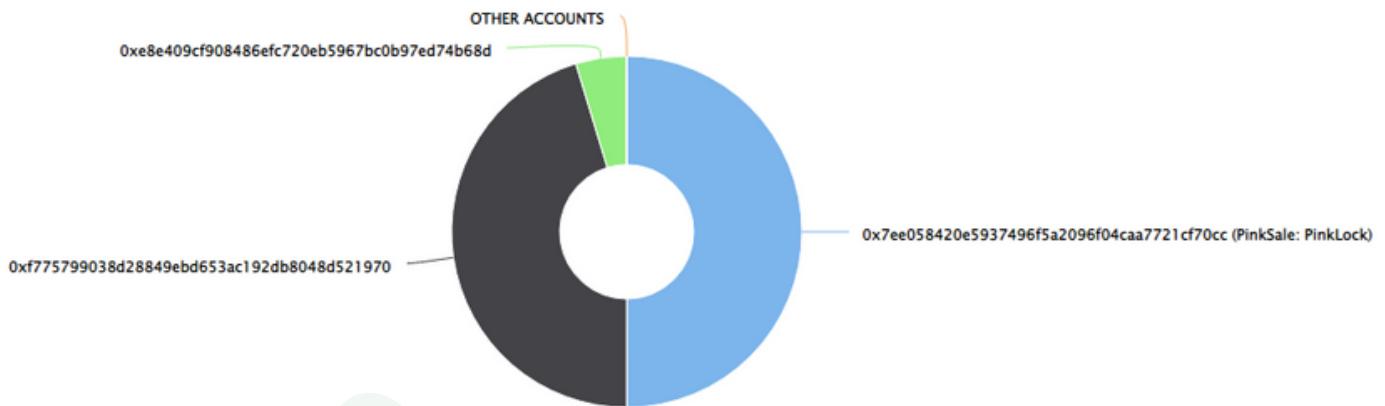


Compiler Check

Final Score: 87.6

SUMMARY

Top 10 holders



Rank	Address	Quantity (Token)	Percentage
1	PinkSale: PinkLock	500,000,000,000	50.0000%
2	0xf775799038d28849ebd653ac192db8048d521970	453,179,999,994.54	45.3180%
3	0xe8e409cf908486efc720eb5967bc0b97ed74b68d	46,820,000,005.46	4.6820%

Conclusion

Project MyLocalPax does not contain any severe issues or risk characteristics. Owner is able to set very high fees and max tx.

SafuAudit has tested the security based on manual and automated tests. Please note that we don't offer any warranties for business model.





SafuAudit.com

