freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

zkLaunchpad

$ZKPAD

03/04/2023

TOKEN OVERVIEW

Fees

• Buy fees:	0%
• Sell fees:	0%

Fees privileges

• Can change buy fees up to 20% and sell fees up to 25%

Ownership

• Owned

Minting

• No mint function

Max Tx Amount / Max Wallet Amount

• Can change max tx amount and max wallet amount (with threshold)

Blacklist

• Blacklist function not detected

Other privileges

• Can exclude / include from fees

TABLE OF CONTENTS

1 DISCLAIMER

2 INTRODUCTION

3 WEBSITE + SOCIALS

4-5 AUDIT OVERVIEW

6-9 OWNER PRIVILEGES

10 CONCLUSION AND ANALYSIS

11 TOKEN DETAILS

12 ZKPAD TOKEN

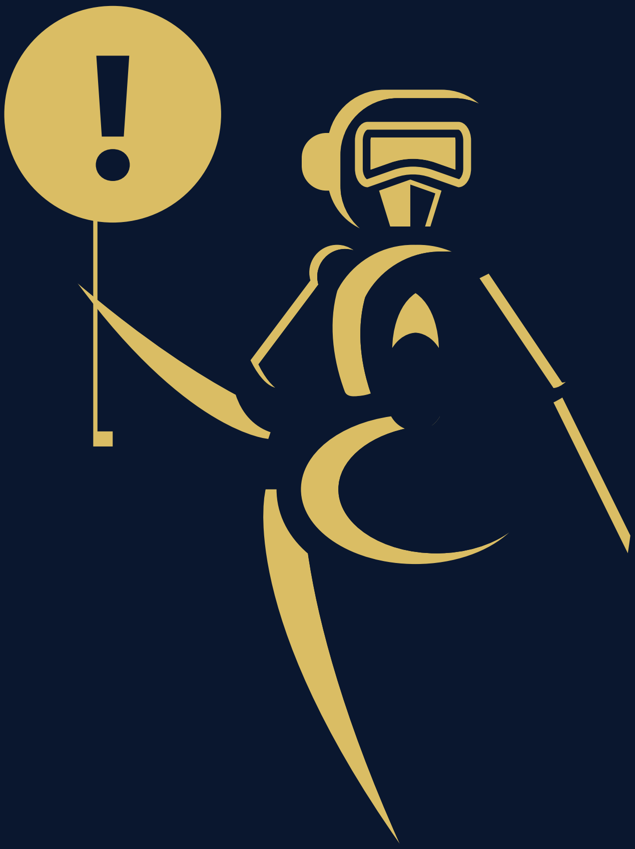
13 TECHNICAL DISCLAIMER

DISCLAIMER

The information provided on this analysis document is only for general information and should not be used as a reason to invest.

FreshCoins Team will take no payment for manipulating the results of this audit.

The score and the result will stay on this project page information on our website https://freshcoins.io
FreshCoins Team does not guarantees that a project will not sell off team supply, or any other scam strategy (RUG or Honeypot etc)



INTRODUCTION

FreshCoins (Consultant) was contracted by
zkLaunchpad (Customer) to conduct a Smart Contract Code Review and Security Analysis.

0x959aB3394246669914BdDEAeB50f8Ac85648615e

Network: zkSync

This report presents the findings of the security assessment of Customer's smart contract and its code review conducted on 03/04/2023

WEBSITE DIAGNOSTIC

https://zklaunchpad.tech/

0-49	50-89	90-100

94	92	95	92	NA
Performance	Accessibility	Best Practices	SEO	Progressive Web App

Socials

Twitter

https://twitter.com/zkPadOfficial

Telegram

https://t.me/zkLaunchpad

AUDIT OVERVIEW

89

Security Score

98 **Static Scan**
Automatic scanning for common vulnerabilities

85 **ERC Scan**
Automatic checks for ERC's conformance

0	High
7	Medium
0	Low
0	Optimizations
0	Informational

No.	Issue description	Checking Status
1	Compiler Errors / Warnings	Passed
2	Reentrancy and Cross-function	Passed
3	Front running	Passed
4	Timestamp dependence	Passed
5	Integer Overflow and Underflow	Passed
6	Reverted DoS	Passed
7	DoS with block gas limit	Passed
8	Methods execution permissions	Passed
9	Exchange rate impact	Passed
10	Malicious Event	Passed
11	Scoping and Declarations	Passed
12	Uninitialized storage pointers	Passed
13	Design Logic	Passed
14	Safe Zeppelin module	Passed

OWNER PRIVILEGES

🟢 Contract owner can't mint tokens after initial contract deploy

🟢 Contract owner can't exclude an address from transactions

🟡 Contract owner can exclude/include wallet from tax

```
function excludeFromFees(address account, bool excluded) public onlyOwner {
        _isExcludedFromFees[account] = excluded;
        emit ExcludeFromFees(account, excluded);
}
```

🟡 Contract owner can exclude/include wallet from tx limitations

```
function excludeFromMaxTransaction(address updAds, bool isEx) public onlyOwner {
        _isExcludedMaxTransactionAmount[updAds] = isEx;
}
```

🟢 Contract owner can change swap settings

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

function updateSwapTokensAtAmount(uint256 newAmount) external onlyOwner returns (bool){
        require(newAmount >= totalSupply() * 1 / 100000, "Swap amount cannot be lower than 0.001% total
supply.");
        require(newAmount <= totalSupply() * 5 / 1000, "Swap amount cannot be higher than 0.5% total
supply.");
        swapTokensAtAmount = newAmount;
        return true;
}
```

🟡 Contract owner has to call enableTrade function to enable trade

(once enabled, can’t be disabled)

```
function enableTrading() external onlyOwner {
        tradingActive = true;
        swapEnabled = true;
}
```

🟡 Contract owner can disable delay between trades

Current value: transferDelayEnabled: true

```
function disableTransferDelay() external onlyOwner returns (bool){
        transferDelayEnabled = false;
        return true;
}
```

● **Contract owner can remove all limits** (tx limitations, wallet limitations, etc)

```
function removeLimits() external onlyOwner returns (bool) {
      limitsInEffect = false;
      return true;
}
```

● **Contract owner can transfer the entire balance of the smart contract to own wallet** (ZKPAD tokens not excluded)

```
function payout() public onlyOwner {
      payable(msg.sender).transfer(address(this).balance);
}
```

● **Contract owner can change buy fees up to 20% and sell fees up to 25%**

```
function updateBuyFees(uint256 _marketingFee, uint256 _liquidityFee, uint256 _devFee) external onlyOwner
{
      buyMarketingFee = _marketingFee;
      buyLiquidityFee = _liquidityFee;
      buyDevFee = _devFee;
      buyTotalFees = buyMarketingFee + buyLiquidityFee + buyDevFee;
      require(buyTotalFees <= 20, "Must keep fees at 20% or less");
}

function updateSellFees(uint256 _marketingFee, uint256 _liquidityFee, uint256 _devFee) external onlyOwner
{
      sellMarketingFee = _marketingFee;
      sellLiquidityFee = _liquidityFee;
      sellDevFee = _devFee;
      sellTotalFees = sellMarketingFee + sellLiquidityFee + sellDevFee;
      require(sellTotalFees <= 25, "Must keep fees at 25% or less");
}
```

● **The liquidity of the contract automatically gets credited into the marketing wallet whenever the 'addLiquidity' function is called inside the contract**

Note that it cannot be called manually but it will be done automatically every time the swap and liquify function is called. Moreover, even after the renouncement of the ownership, this liquidity will still be credited to the marketing wallet

```
function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {
      _approve(address(this), address(uniswapV2Router), tokenAmount);

      uniswapV2Router.addLiquidityETH{value: ethAmount}(
         address(this),
         tokenAmount,
         0, // slippage is unavoidable
         0, // slippage is unavoidable
         marketingWallet,
         block.timestamp,
         0,
         false
      );
}
```

Contract owner can change tx limitations and wallet limitations (with threshold)

```
function updateMaxTxnAmount(uint256 newNum) external onlyOwner {
      require(newNum >= (totalSupply() * 1 / 1000)/1e18, "Cannot set maxTransactionAmount lower than
0.1%");
      maxTransactionAmount = newNum * (10**18);
}

function updateMaxWalletAmount(uint256 newNum) external onlyOwner {
      require(newNum >= (totalSupply() * 5 / 1000)/1e18, "Cannot set maxWallet lower than 0.5%");
      maxWallet = newNum * (10**18);
}
```

Contract owner can change marketingWallet and devWallet addresses

Current values:

marketingWallet : 0xD0Cb58aB0E936540bDf2127C7C3cDf1524216D12

devWallet : 0xD0Cb58aB0E936540bDf2127C7C3cDf1524216D12

```
function updateMarketingWallet(address newMarketingWallet) external onlyOwner {
      emit marketingWalletUpdated(newMarketingWallet, marketingWallet);
      marketingWallet = newMarketingWallet;
}

function updateDevWallet(address newWallet) external onlyOwner {
      emit devWalletUpdated(newWallet, devWallet);
      devWallet = newWallet;
}
```

Contract owner can transfer ownership

```
function transferOwnership(address newOwner) public virtual onlyOwner {
      require(newOwner != address(0), "Ownable: new owner is the zero address");
      emit OwnershipTransferred(_owner, newOwner);
      _owner = newOwner;
}
```

Contract owner can renounce ownership

```
function renounceOwnership() public virtual onlyOwner {
      emit OwnershipTransferred(_owner, address(0));
      _owner = address(0);
}
```

Recommendation:

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. The risk can be prevented by temporarily locking the contract or renouncing ownership.

CONCLUSION AND ANALYSIS

Smart Contracts within the scope were manually reviewed and analyzed with static tools.

Audit report overview contains all found security vulnerabilities and other issues in the reviewed code.

Found no HIGH issues during the first review.

TOKEN DETAILS

Details

Buy fees:	0%
Sell fees:	0%
Max TX:	150,000
Max Sell:	N/A

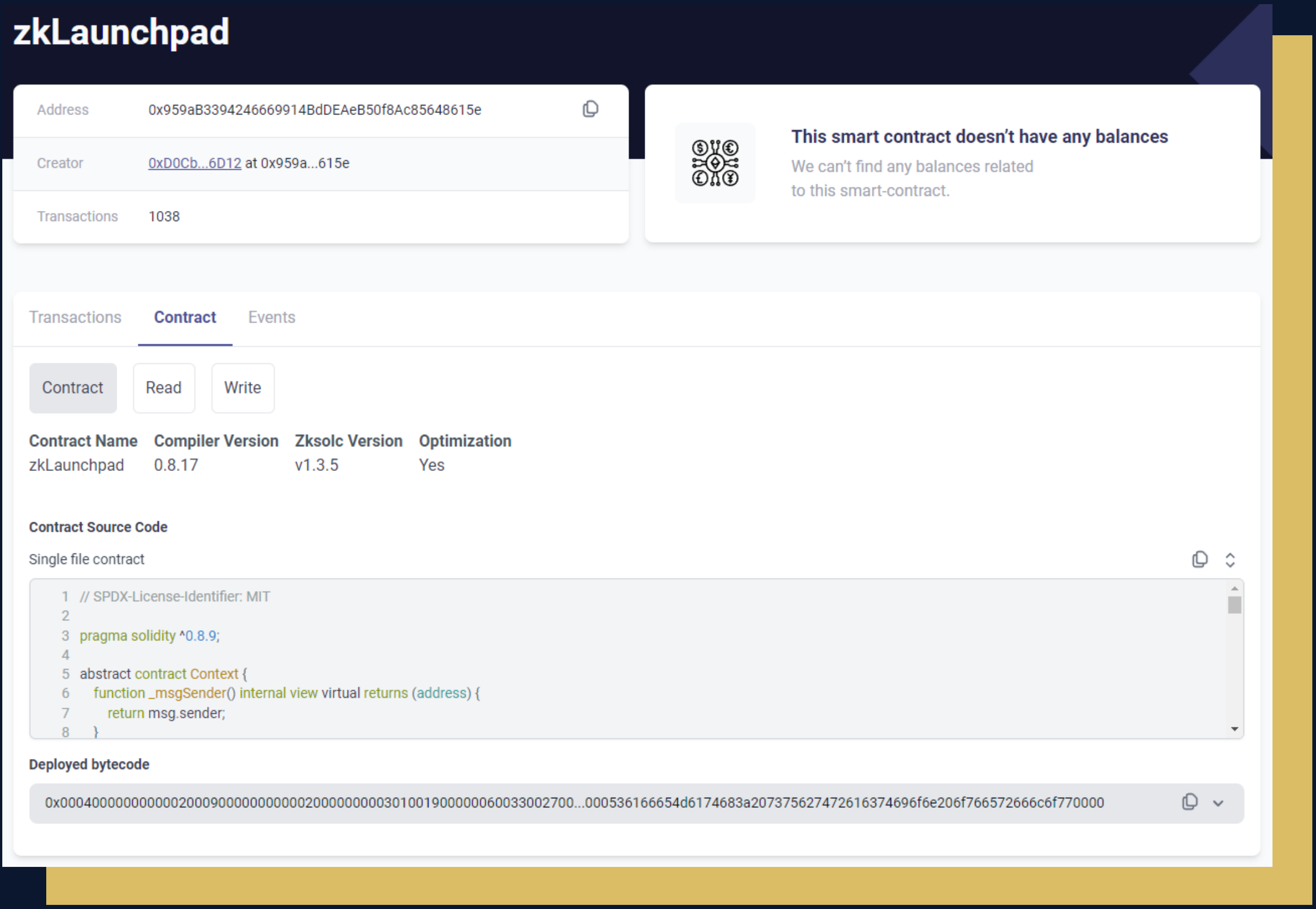
Honeypot Risk

Ownership:	Owned
Blacklist:	Not detected
Modify Max TX:	Detected
Modify Max Sell:	Not detected
Disable Trading:	Not detected

Rug Pull Risk

Liquidity:	N/A
Holders:	Clear

ZKPAD TOKEN



zkLaunchpad

Address	0x959aB3394246669914BdDEAeB50f8Ac85648615e
Creator	0xD0Cb...6D12 at 0x959a...615e
Transactions	1038

This smart contract doesn't have any balances
We can't find any balances related to this smart-contract.

Contract Name	Compiler Version	Zksolc Version	Optimization
zkLaunchpad	0.8.17	v1.3.5	Yes

Contract Source Code

Single file contract

```
// SPDX-License-Identifier: MIT

pragma solidity ^0.8.9;

abstract contract Context {
    function _msgSender() internal view virtual returns (address) {
        return msg.sender;
    }
```

Deployed bytecode

0x0004000000000000020009000000000000020000000003010019000000060033002700...000536166654d6174683a2073756274726163374696f6e206f766572666c6f770000

TECHNICAL DISCLAIMER

Smart contracts are deployed and executed on the blockchain platform. The platform, its programming language, and other software related to the smart contract can have its vulnerabilities that can lead to hacks. The audit can't guarantee the explicit security of the audited project / smart contract.