



SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT



Crypddolphin
\$CPD

10/05/2023

TOKEN OVERVIEW

Fees

- Buy fees: 3%
- Sell fees: 5%

Fees privileges

- Can change fees up to 100%

Ownership

- Owned

Minting

- No mint function

Max Tx Amount / Max Wallet Amount

- Can change max tx amount (without 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

CPD TOKEN ANALYTICS &
TOP 10 TOKEN HOLDERS

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 **Crypddolphin** (Customer) to conduct a Smart Contract Code Review and Security Analysis.

0xD2Ea0850d0169260B1c89Ab880671e7Dc5D97184

Network: **Binance Smart Chain (BSC)**

This report presents the findings of the security assessment of Customer's smart contract and its code review conducted on **10/05/2023**



WEBSITE DIAGNOSTIC

<https://www.crypddolphin.io/>



0-49



50-89



90-100



Performance



Accessibility



Best
Practices



SEO



Progressive
Web App

Socials



Twitter

<https://twitter.com/Crypddolphi23488>



Telegram

https://t.me/CPD_EN

AUDIT OVERVIEW



Security Score
HIGH RISK
Audit FAIL



Static Scan
Automatic scanning for
common vulnerabilities



ERC Scan
Automatic checks for
ERC's conformance



High



Medium



Low



Optimizations



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 wallet from tax**

```
function updateExcludedFromFees(address _address, bool state) external onlyOwner {
    excludedFromFees[_address] = state;
}
```

- **Contract owner can change swap settings (without threshold)**

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

function setSwapThreshold(uint256 new_amount) external onlyOwner {
    swapThreshold = new_amount * 10**decimals();
}
```

- **Contract owner has to call `confirmLpFilled` function to enable trade**

Note that any wallet excluded from fees can trade even if trading is disabled

```
function confirmLpFilled() external onlyOwner{
    lpFilled = true;
    swapEnabled = true;
}
.
.
.
require(lpFilled || excludedFromFees[sender] || excludedFromFees[recipient], "Trading disabled");
.
.
.
```

- **Contract owner can change max tx amount (without threshold)**

Note that setting the value too low may prevent users from making purchase transactions

```
function setMaxBuy(uint256 amount) external onlyOwner{
    maxBuy = amount * 10**decimals();
}
```

● Contract owner can change fees up to 100%

```
function setBuyTaxes(uint256 _marketing, uint256 _dev, uint256 _liquidity) external onlyOwner{
    buyTaxes = Taxes(_marketing, _dev, _liquidity);
    totalBuyTax = _marketing + _dev + _liquidity;
}

function setSellTaxes(uint256 _marketing, uint256 _dev, uint256 _liquidity) external onlyOwner{
    sellTaxes = Taxes(_marketing, _dev, _liquidity);
    totalSellTax = _marketing + _dev + _liquidity;
}
```

● Contract owner can change marketingWallet and devFunds addresses

Current values:

marketingWallet : 0x5b5889569770df361c52f1de74ced521a6cc2959

devFunds : 0x83140ed09759fe27a2566363be5b342e4da96758

```
function updateDevFunds(address newWallet) external onlyOwner{
    devFunds = newWallet;
}

function updateMarketingWallet(address newWallet) external onlyOwner{
    marketingWallet = newWallet;
}
```

● Contract owner can withdraw tokens from smart contract

Native tokens excluded

```
function rescueBEP20(address tokenAddress, uint256 amount) external onlyOwner{
    require(tokenAddress != address(this), "Can't take self token");
    IERC20(tokenAddress).transfer(owner(), amount);
}

function rescueBNB(uint256 weiAmount) external onlyOwner{
    payable(owner()).sendValue(weiAmount);
}
```

● Contract owner can transfer ownership

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

function _transferOwnership(address newOwner) internal virtual {
    address oldOwner = _owner;
    _owner = newOwner;
    emit OwnershipTransferred(oldOwner, newOwner);
}
```

- The liquidity of the contract automatically gets credited into the owner's 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 owner's wallet.

```
function addLiquidity(uint256 tokenAmount, uint256 bnbAmount) private {  
    // approve token transfer to cover all possible scenarios  
    _approve(address(this), address(router), tokenAmount);  
  
    // add the liquidity  
    router.addLiquidityETH{value: bnbAmount}(  
        address(this),  
        tokenAmount,  
        0, // slippage is unavoidable  
        0, // slippage is unavoidable  
        owner(),  
        block.timestamp  
    );  
}
```

- Contract owner can renounce ownership

```
function renounceOwnership() public virtual onlyOwner {  
    _transferOwnership(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 3 HIGH issues during the first review.

TOKEN DETAILS

Details

Buy fees:	3%
Sell fees:	5%
Max TX:	0
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:	Clean



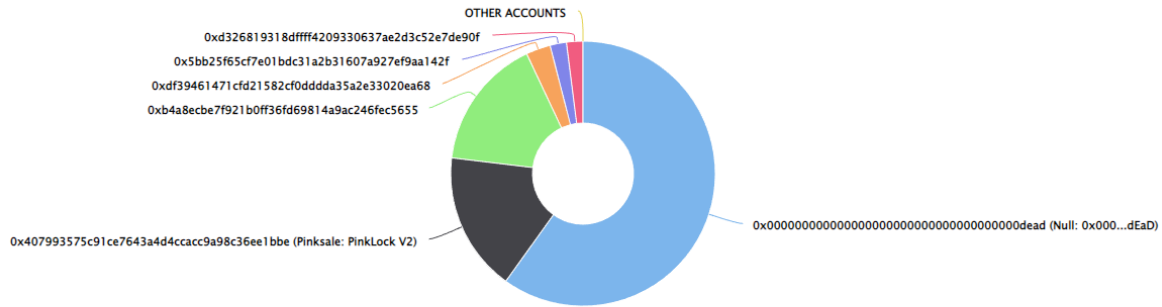
CPD TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

The top 10 holders collectively own 100.00% (100,000,000,000.00 Tokens) of Crypdolphin

Token Total Supply: 100,000,000,000.00 Token | Total Token Holders: 6

Crypdolphin Top 10 Token Holders

Source: BscScan.com



(A total of 100,000,000,000.00 tokens held by the top 10 accounts from the total supply of 100,000,000,000.00 token)

Rank	Address	Quantity (Token)	Percentage
1	Null: 0x000...dEaD	59,920,000,160.8	59.9200%
2	Pinksale: PinkLock V2	17,000,000,000	17.0000%
3	0xb4a8ecbe7f921b0ff36fd69814a9ac246fec5655	16,079,999,839.2	16.0800%
4	0xdf39461471cfd21582cf0ddda35a2e33020ea68	3,000,000,000	3.0000%
5	0x5bb25f65cf7e01bdc31a2b31607a927ef9aa142f	2,000,000,000	2.0000%
6	0xd326819318dffff4209330637ae2d3c52e7de90f	2,000,000,000	2.0000%

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.

