expelee

Building the Futuristic Blockchain Ecosystem

SECURITY AUDIT REPORT

Candle Catcher

TABLE OF CONTENTS

OVERVIEW

The Expelee team has performed a line-by-line manual analysis and automated review of the smart contract. The smart contract was analysed mainly for common smart contract vulnerabilities, exploits, and manipulation hacks. According to the smart contract audit:

Audit Result	**Passed**
KYC Verification	**-**
Audit Date	**21 April 2023**

PROJECT DESCRIPTION

Candle Catcher is also known as Candle thief as we steal candles from others chart and place it in ours to make it green. We aim to create a unique and sustainable environment for our users by leveraging a distinctive approach to tokenomics and investment strategy.

SOCIAL MEDIA PROFILES

CANDLE CATCHER

https://t.me/CandleCatcherOfficial

https://twitter.com/CandleCatcher

https://candlecatchercoin.com/

It's always good to check the social profiles of the project, before making your investment.

Team Expelee

CONTRACT DETAILS

Token Name: Candle Catcher

Symbol: CNDL

Network: Binance Smart Chain

Language: Solidity

Contract Address :
0x492AAbEB7cF62BFD7EC95f1094fC696bB4DDa96a

Total Supply: 1,000,000,000,000,000

Contract SHA-256 Checksum:
ae2ccb14ec54f1265b4c1e70fcf67e088586dcd1

Owner's Wallet:

0x2f6db8d76be994bc84ead5a24ef35de84b2a3e45

Testnet:
https://testnet.bscscan.com/token/0x9089D4548335dA75ACd9F097f6764C1A7Ba9Cc46

OWNER PRIVILEGES

- Contract owner is not able to change buy/sell fees (8% each)
- Contract owner is not able to set a transfer fee (0% transfer fee)
- Contract owner is not able to set limits for buy/sell/transfer amounts
- Contract owner is not able to blacklist an arbitrary wallet
- Contract owner is not able to disable trades/transfers
- Contract owner is not able to mint new tokens

AUDIT METHODOLOGY

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

- DE
- Open Zeppelin
- Code Analyzer
- Solidity Code
- Compiler
- Hardhat

VULNERABILITY CHECKS

Design Logic	Passed
Compiler warnings	Passed
Private user data leaks	Passed
Timestamps dependence	Passed
Integer overflow and underflow	Passed
Race conditions & reentrancy. Cross-function race conditions	Passed
Possible delays in data delivery	Passed
Oracle calls	Passed
Front Running	Passed
DoS with Revert	Passed
DoS with block gas limit	Passed
Methods execution permissions	Passed
Economy model	Passed
Impact of the exchange rate on the logic	Passed
Malicious event log	Passed
Scoping and declarations	Passed
Uninitialized storage pointers	Passed
Arithmetic accuracy	Passed
Cross-function race conditions	Passed
Safe Zepplin module	Passed

RISK CLASSIFICATION

When performing smart contract audits, our specialists look for known vulnerabilities as well as logical and acces control issues within the code. The exploitation of these issues by malicious actors may cause serious financial damage to projects that failed to get an audit in time. We categorize these vulnerabilities by the following levels:

High Risk

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

Medium Risk

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

Low Risk

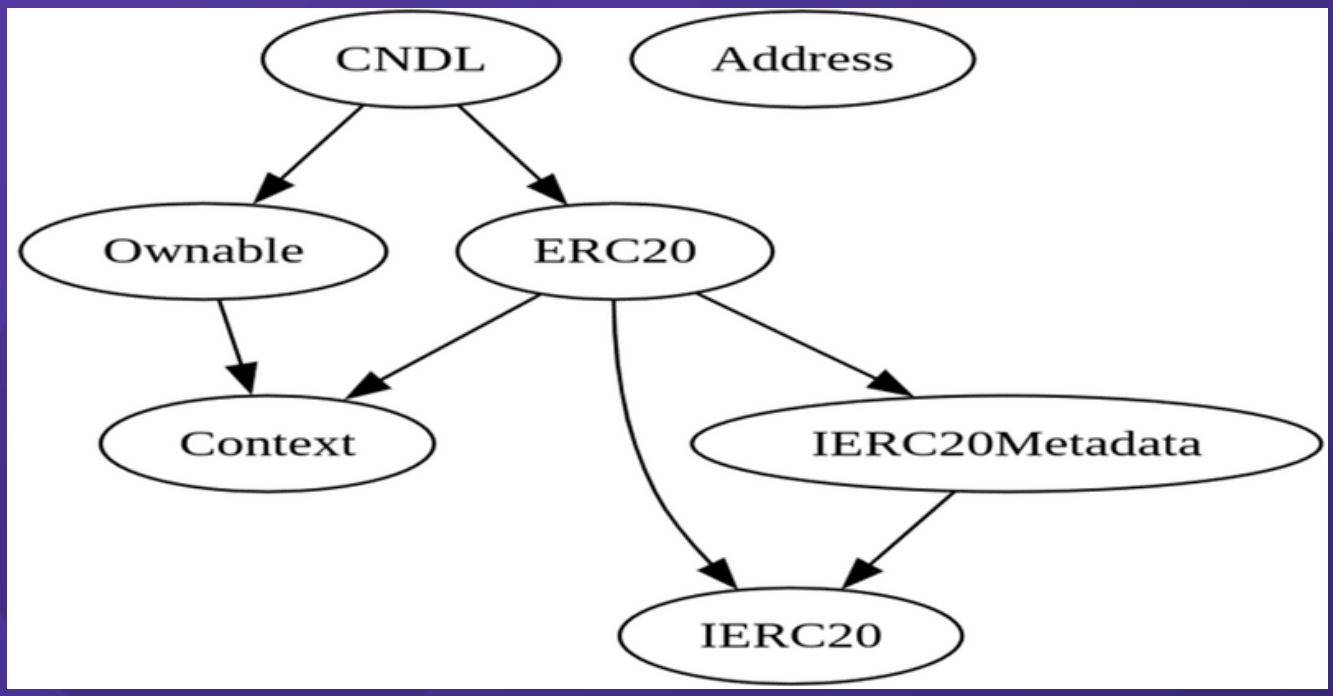
Issues on this level are minor details and warning that can remain unfixed.

Informational

Issues on this level are minor details and warning that can remain unfixed.

expelee

INHERITANCE TREES



expelee

FUNCTION DETAILS

Symbol	Meaning
🔴	Function can modify state
💵	Function is payable

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
IUniswapV2Factory	Interface			
└	feeTo	External ❗	NO ❗	
└	feeToSetter	External ❗	NO ❗	
└	getPair	External ❗	NO ❗	
└	allPairs	External ❗	NO ❗	
└	allPairsLength	External ❗	NO ❗	
└	createPair	External ❗	🔴 NO ❗	
└	setFeeTo	External ❗	🔴 NO ❗	
└	setFeeToSetter	External ❗	🔴 NO ❗	
IUniswapV2Pair	Interface			
└	name	External ❗	NO ❗	
└	symbol	External ❗	NO ❗	
└	decimals	External ❗	NO ❗	
└	totalSupply	External ❗	NO ❗	
└	balanceOf	External ❗	NO ❗	
└	allowance	External ❗	NO ❗	
└	approve	External ❗	🔴 NO ❗	
└	transfer	External ❗	🔴 NO ❗	
└	transferFrom	External ❗	🔴 NO ❗	
└	DOMAIN_SEPARATOR	External ❗	NO ❗	
└	PERMIT_TYPEHASH	External ❗	NO ❗	
└	nonces	External ❗	NO ❗	
└	permit	External ❗	🔴 NO ❗	
└	MINIMUM_LIQUIDITY	External ❗	NO ❗	
└	factory	External ❗	NO ❗	
└	token0	External ❗	NO ❗	
└	token1	External ❗	NO ❗	
└	getReserves	External ❗	NO ❗	
└	price0CumulativeLast	External ❗	NO ❗	
└	price1CumulativeLast	External ❗	NO ❗	
└	kLast	External ❗	NO ❗	
└	mint	External ❗	🔴 NO ❗	
└	burn	External ❗	🔴 NO ❗	
└	swap	External ❗	🔴 NO ❗	
└	skim	External ❗	🔴 NO ❗	
└	sync	External ❗	🔴 NO ❗	
└	initialize	External ❗	🔴 NO ❗	
IUniswapV2Router01	Interface			
└	factory	External ❗	NO ❗	
└	WETH	External ❗	NO ❗	
└	addLiquidity	External ❗	🔴 NO ❗	
└	addLiquidityETH	External ❗	💵 NO ❗	
└	removeLiquidity	External ❗	🔴 NO ❗	
└	removeLiquidityETH	External ❗	🔴 NO ❗	

expelee

FUNCTION DETAILS

└	removeLiquidityWithPermit	External ❗	🔴	NO ❗
└	removeLiquidityETHWithPermit	External ❗	🔴	NO ❗
└	swapExactTokensForTokens	External ❗	🔴	NO ❗
└	swapTokensForExactTokens	External ❗	🔴	NO ❗
└	swapExactETHForTokens	External ❗	💵	NO ❗
└	swapTokensForExactETH	External ❗	🔴	NO ❗
└	swapExactTokensForETH	External ❗	🔴	NO ❗
└	swapETHForExactTokens	External ❗	💵	NO ❗
└	quote	External ❗		NO ❗
└	getAmountOut	External ❗		NO ❗
└	getAmountIn	External ❗		NO ❗
└	getAmountsOut	External ❗		NO ❗
└	getAmountsIn	External ❗		NO ❗
IUniswapV2Router02	Interface	IUniswapV2Router01		
└	removeLiquidityETHSupportingFeeOnTransferTokens	External ❗	🔴	NO ❗
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🔴	NO ❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🔴	NO ❗
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO ❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🔴	NO ❗
IERC20	Interface			
└	totalSupply	External ❗		NO ❗
└	balanceOf	External ❗		NO ❗
└	transfer	External ❗	🔴	NO ❗
└	allowance	External ❗		NO ❗
└	approve	External ❗	🔴	NO ❗
└	transferFrom	External ❗	🔴	NO ❗
IERC20Metadata	Interface	IERC20		
└	name	External ❗		NO ❗
└	symbol	External ❗		NO ❗
└	decimals	External ❗		NO ❗
Address	Library			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🔴	
└	functionCall	Internal 🔒	🔴	
└	functionCall	Internal 🔒	🔴	
└	functionCallWithValue	Internal 🔒	🔴	
└	functionCallWithValue	Internal 🔒	🔴	
└	functionStaticCall	Internal 🔒		
└	functionStaticCall	Internal 🔒		
└	functionDelegateCall	Internal 🔒	🔴	
└	functionDelegateCall	Internal 🔒	🔴	
└	verifyCallResultFromTarget	Internal 🔒		
└	verifyCallResult	Internal 🔒		
└	_revert	Private 🔐		
Context	Implementation			
└	_msgSender	Internal 🔒		

expelee

FUNCTION DETAILS

└	_msgData	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO ❗
└	owner	Public ❗		NO ❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
ERC20	Implementation	Context, IERC20, IERC20Metadata		
└	<Constructor>	Public ❗	🛑	NO ❗
└	name	Public ❗		NO ❗
└	symbol	Public ❗		NO ❗
└	decimals	Public ❗		NO ❗
└	totalSupply	Public ❗		NO ❗
└	balanceOf	Public ❗		NO ❗
└	transfer	Public ❗	🛑	NO ❗
└	allowance	Public ❗		NO ❗
└	approve	Public ❗	🛑	NO ❗
└	transferFrom	Public ❗	🛑	NO ❗
└	increaseAllowance	Public ❗	🛑	NO ❗
└	decreaseAllowance	Public ❗	🛑	NO ❗
└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	
CNDL	Implementation	ERC20, Ownable		
└	<Constructor>	Public ❗	🛑	ERC20
└	<Receive Ether>	External ❗	💵	NO ❗
└	claimStuckTokens	External ❗	🛑	onlyOwner
└	excludeFromFees	External ❗	🛑	onlyOwner
└	isExcludedFromFees	Public ❗		NO ❗
└	changeMarketingWallet	External ❗	🛑	onlyOwner
└	enableTrading	External ❗	🛑	onlyOwner
└	_transfer	Internal 🔒	🛑	
└	setSwapEnabled	External ❗	🛑	onlyOwner
└	setSwapTokensAtAmount	External ❗	🛑	onlyOwner
└	swapAndLiquify	Private 🔐	🛑	
└	swapAndSendMarketing	Private 🔐	🛑	

MANUAL REVIEW

Severity Criteria

Expelee assesses the severity of disclosed vulnerabilities according to methodology based on OWASP standarts.

Vulnerabilities are dividend into three primary risk categroies:
High
Medium
Low

High-level considerations for vulnerabilities span the following key areas when conducting assessments:

- Malicious input handling
- Escalation of privileges
- Arithmetic
- Gas use

Overall Risk Severity				
Impact	HIGH	Medium	High	Critical
	MEDIUM	Low	Medium	High
	LOW	Note	Low	Medium
		LOW	MEDIUM	HIGH
	Likelihood			

FINDINGS

Findings	Severity	Found
High Risk	High	1 (1 Resolved)
Medium Risk	Medium	0
Low Risk	Low	0
Suggestion & discussion	Informational	0
Gas Optimizations	Gas Opt.	0

HIGH RISK FINDING

Enabling trades is not guaranteed

Severity : High

Overview

The owner of the contract must enable trades for public, otherwise no one would be able to buy/sell/transfer their tokens except whitelisted wallets.

```
  function enableTrading() external onlyOwner {
require(!tradingEnabled, "Trading already enabled.");
tradingEnabled = true;
swapEnabled = true;
  }
```

Suggestion :

To mitigate this issue there are several options:
- Temporary transfer ownership of the contract to a pinksale safu developer (done)
- Enable tradings before presale
- Allowing enabling of the trades after a specifiec time by any token holder

Example:

```
  function enableTrading() external {
            require(msg.sender == owner || elapsedTimeSincePresale >
threshold, “Caller                must be owner, or threshold must be
passed’);
pairAddress = _pairAddress;
tradingStatus = true;
 }
```

HIGH RISK FINDING

Issue Status: Resolved

Contract is owned and developed by pinksale safu developer (coinsult), hence enabling trades is guaranteed.

ABOUT EXPELEE

Expelee is a product-based aspirational Web3 start-up. Coping up with numerous solutions for blockchain security and constructing a Web3 ecosystem from deal making platform to developer hosting open platform, while also developing our own commercial and sustainable blockchain.

www.expelee.com

expeleeofficial

expelee

Expelee

expelee

expelee_official

expelee-co

expelee

Building the Futuristic Blockchain Ecosystem

DISCLAIMER

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment. Team provides no guarantess against the sale of team tokens or the removal of liquidity by the project audited in this document.

Always do your own research and project yourselves from being scammed. The Expelee team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools.

Under no circumstances did Expelee receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Alway do your own research and protect yourselves from scams.

This document should not be presented as a reason to buy or not buy any particular token. The Expelee team disclaims any liability for the resulting losses.