expelee

A Secure Place For Web3

SMART CONTRACT AUDIT OF

ProspectorCrane Presale

Contract Address

0xa1fb785eC5D0d8A06C32B8d60dF7E2F5e3C62650

Audit Summary

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**

Ownership: NOT **RENOUNCED**

KYC Verification: Not Done

Audit Date: 04/08/2022

Audit Team: **EXPELEE**

Be aware that smart contracts deployed on the blockchain aren't resistant to internal exploit, external vulnerability, or hack. For a detailed understanding of risk severity, source code vulnerability, functional hack, and audit disclaimer, kindly refer to the audit.

DISCLAMER

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 guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document.

Always Do your own research and protect 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. Always 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.

Contract Review

Contract Name	**CoinToken**
Compiler Version	**v0.8.12+commit.f00d7308**
Optimization	**Yes with 200 runs**
License	**None License**
Explorer	**https://bscscan.com/address/0xa1fb785eC5D0d8A06C32B8d60dF7E2F5e3C62650#code**
Symbol	**CRANES**
Decimals	**18**
Total Supply	**500,000**
Domain	**https://www.prospectornft.com/**

Project Review

Token Name: ProspectorCrane

Web Site: https://www.prospectornft.com/

Twitter: https://twitter.com/ProspectorCrane

Telegram: https://t.me/ProspectorNFT_officia

Contract Address:
0xa1fb785eC5D0d8A06C32B8d60dF7E2F5e3C62650

Platform: Binance Smart Chain

Token Type: BEP 20

Language: SOLIDITY

Audit Methodology

The scope of this report is to audit the smart contract source code. We have scanned the contract and reviewed the project for common vulnerabilities, exploits, hacks, and back-doors. Below is the list of commonly known smart contract vulnerabilities, exploits, and hacks:

Category

Smart Contract Vulnerabilities	- Unhandled Exceptions - Transaction Order Dependency - Integer Overflow - Unrestricted Action - Incorrect Inheritance Order - Typographical Errors - Requirement Violation
Source Code Review	- Gas Limit and Loops - Deployment Consistency - Repository Consistency - Data Consistency - Token Supply Manipulation
Functional Assessment	- Operations Trail & Event Generation - Assets Manipulation - Liquidity Access

Vulnerability Checklist

№	Description.	Result
1	Compiler warnings.	Passed
2	Race conditions and Re-entrancy. Cross-function raceconditions.	Passed
3	Possible delays in data delivery.	Passed
4	Oracle calls.	Passed
5	Front running.	Passed
6	Timestamp dependence.	Passed
7	Integer Overflow and Underflow.	Passed
8	DoS with Revert.	Passed
9	DoS with block gas limit.	Passed
10	Methods execution permissions.	Passed
11	Economy model.	Passed
12	The impact of the exchange rate on the logic.	Passed
13	Private user data leaks.	Passed
14	Malicious Event log.	Passed
15	Scoping and Declarations.	Passed
16	Uninitialized storage pointers.	Passed
17	Arithmetic accuracy.	Passed
18	Design Logic.	Passed
19	Cross-function race conditions.	Passed
20	Safe Zeppelin module.	Passed
21	Fallback function security.	Passed

Manual Audit

Low-Risk

4 low-risk code issues found

Medium-Risk

0 medium-risk code issues found

High-Risk

0 high-risk code issues found

Audit Summary

Number of lines: 1695 (+ 0 in dependencies, + 0 in tests)

Number of assembly lines: 0

Number of contracts: 18 (+ 0 in dependencies, + 0 tests)

Number of optimization issues: 31

Number of informational issues: 53

Number of low issues: 4

Number of medium issues: 0

Number of high issues: 0

ERCs: ERC20, ERC2612

Name	# functions	ERCS	ERC20 info	Complex code	Features
SafeMath	8			No	
SafeMathInt	7			No	
SafeMathUint	1			No	
Clones	4			No	Assembly
IUniswapV2Router02	24			No	Receive ETH
IUniswapV2Factory	8			No	
IUniswapV2Pair	26	ERC20,ERC2612	No Minting Approve Race Cond.	No	
TokenDividendTracker	72	ERC20	No Minting Approve Race Cond.	Yes	Tokens interaction
CoinToken	71	ERC20	No Minting Approve Race Cond.	Yes	Receive ETH Send ETH Tokens interaction

Low-Risk

1) Functions that send Ether to arbitrary destinations

Unprotected call to a function sending Ether to an arbitrary address.

```
function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {
    // approve token transfer to cover all possible scenarios
    _approve(address(this), address(uniswapV2Router), tokenAmount);
    // add the liquidity
    uniswapV2Router.addLiquidityETH{value: ethAmount}(
        address(this),
        tokenAmount,
        0, // slippage is unavoidable
        0, // slippage is unavoidable
        _node,
        block.timestamp
    );

}
```

Recommendation

Ensure that an arbitrary user cannot withdraw unauthorized funds.

2) Reentrancy vulnerabilities

Detection of the reentrancy bug. Do not report reentrancies that don't involve Ether (see reentrancy-no-eth)

```
uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
    tokenAmount,
    0, // accept any amount of ETH
    path,
    address(this),
    block.timestamp
);
```

Recommendation

Apply the check-effects-interactions pattern.

3) Unchecked transfer

The return value of an external transfer/transferFrom call is not checked.

```
    function swapAndSendToFee(uint256 tokens) private  {
    uint256 initialCAKEBalance = IERC20(rewardToken).balanceOf(address(this));
    swapTokensForCake(tokens);
    uint256 newBalance = (IERC20(rewardToken).balanceOf(address(this))).sub(initialCAKEBalance);
    IERC20(rewardToken).transfer(_marketingWalletAddress, newBalance);
    AmountMarketingFee = AmountMarketingFee - tokens;
}
```

Recommendation

Use SafeERC20, or ensure that the transfer/transferFrom return value is checked.

4) Missing events arithmetic

Detect missing events for critical arithmetic parameters.

```
      function setBuyTaxes(uint256 liquidity, uint256 rewardsFee, uint256 marketingFee, uint256
deadFee) external onlyOwner {

    require(rewardsFee.add(liquidity).add(marketingFee).add(deadFee) <= 25, "Total buy fee is over
25%");
    buyTokenRewardsFee = rewardsFee;
    buyLiquidityFee = liquidity;
    buyMarketingFee = marketingFee;
    buyDeadFee = deadFee;

}
```

Recommendation

Emit an event for critical parameter changes.

Manual Audit (Contract Function)

```
contract CoinToken is ERC20, Ownable {
    using SafeMath for uint256;

    IUniswapV2Router02 public uniswapV2Router;
    address public  uniswapV2Pair;

    bool private swapping;

    TokenDividendTracker public dividendTracker;

    address public rewardToken;

    uint256 public swapTokensAtAmount;

    uint256 public buyTokenRewardsFee;
    uint256 public sellTokenRewardsFee;
    uint256 public buyLiquidityFee;
    uint256 public sellLiquidityFee;
    uint256 public buyMarketingFee;
    uint256 public sellMarketingFee;
    uint256 public buyDeadFee;
    uint256 public sellDeadFee;
    uint256 public AmountLiquidityFee;
    uint256 public AmountTokenRewardsFee;
    uint256 public AmountMarketingFee;

    address public _marketingWalletAddress;
    address private _node;

    address public deadWallet = 0x000000000000000000000000000000000000dEaD;
    mapping(address => bool) public _isEnemy;

    uint256 public gasForProcessing;

     // exlcude from fees and max transaction amount
    mapping (address => bool) private _isExcludedFromFees;

    // store addresses that a automatic market maker pairs. Any transfer *to* these addresses
    // could be subject to a maximum transfer amount
    mapping (address => bool) public automatedMarketMakerPairs;

    event UpdateDividendTracker(address indexed newAddress, address indexed oldAddress)
```

Important Points To Consider

✓ Verified contract source

✓ Token is sellable (not a honeypot) at this time

✗ Ownership renounced or source does not contain an owner contract

✗ Source does not contain a fee modifier

✗ Buy fee is less than 5%(9%)

✗ Sell fee is less than 5%(9%)

✓ Owner/creator wallet contains less than 5% of circulating token supply (8.75%)

✓ All other holders possess less than 5% of circulating token supply.

About Expelee

Expelee is a community driven organisation dedicated to fostering an anti-rug movement. We're here to keep investment safe from fraudsters. We've encountered several rug pulls and know how it feels to be duped, which is why we don't want anybody else to go through the same experience. We are here to raise awareness through our services so that the future of cryptocurrency can be rug-free.

The auditing process focuses to the following considerations with collaboration of an expert team:

- Functionality test of the Smart Contract to determine if proper logic has been followed throughout the whole process.
- Manually detailed examination of the code line by line by experts.
- Live test by multiple clients using Test net.
- Analysing failure preparations to check how the Smart
- Contract performs in case of any bugs and vulnerabilities.
- Checking whether all the libraries used in the code are on the latest version.
- Analysing the security of the on-chain data.

Social Media

/Expelee

/ExpeleeOfficial

/expelee-co