Coinsult

Advanced Manual Smart Contract Audit

Project: Smart Lottery

Website: http://smartlotterydefi.io

● Low-Risk	● Medium-Risk	● High-Risk
4 low-risk code issues found	0 medium-risk code issues found	0 high-risk code issues found

Contract Address

0xfEA099BB53daF7Bb7248042e53983d8715816Aa4

Disclaimer

Coinsult is not responsible if a project turns out to be a scam, rug-pull or honeypot. We only provide a detailed analysis for your own research.

Coinsult is not responsible for any financial losses. Nothing in this contract audit is financial advice, please do your own research.

The information provided in this audit is for informational purposes only and should not be considered investment advice. Coinsult does not endorse, recommend, support or suggest to invest in any project.

Coinsult can not be held responsible for when a project turns out to be a rug-pull, honeypot or scam.

Tokenomics

Rank	Address	Quantity (Token)	Percentage
1	0x09c0e1fac50d6ec7d2df142257213d881bf2fb2b	100,000,000	100.0000%

Source Code

Coinsult was comissioned by Smart Lottery to perform an audit based on the following smart contract:

https://bscscan.com/address/0xfEA099BB53daF7Bb7248042e53983d8715816Aa4#code

Warning: Audit partner notified us that this project is likely to be related to Luck2Earn, which resulted in a scam. Please DYOR when investing in this project.

❌ KYC failed.

Manual Code Review

In this audit report we will highlight all these issues:

🟢 Low-Risk

4 low-risk code
issues found

🟡 Medium-Risk

0 medium-risk code
issues found

🔴 High-Risk

0 high-risk code
issues found

The detailed report continues on the next page...

● **Low-Risk:** Could be fixed, will not bring problems.

Contract contains Reentrancy vulnerabilities

Additional information: This combination increases risk of malicious intent. While it may be justified by some complex mechanics (e.g. rebase, reflections, buyback).

More information: Slither

```
External calls:
- swapAndLiquify(contractTokenBalance) (smart.sol#267)
- idexV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timesta
- idexV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),b
External calls sending eth:
- swapAndLiquify(contractTokenBalance) (smart.sol#267)
- recipient.transfer(amount) (smart.sol#232)
- idexV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timesta
State variables written after the call(s):
- _balances[sender] = _balances[sender].sub(amount,Insufficient Balance) (smart.sol#270)
- _balances[recipient] = _balances[recipient].add(finalAmount) (smart.sol#275)
- finalAmount = takeFee(sender,recipient,amount) (smart.sol#272-273)
- _balances[address(this)] = _balances[address(this)].add(feeAmount) (smart.sol#371)
```

Recommendation

Apply the check-effects-interactions pattern.

Exploit scenario

```
function withdrawBalance(){
    // send userBalance[msg.sender] Ether to msg.sender
    // if mgs.sender is a contract, it will call its fallback function
    if( ! (msg.sender.call.value(userBalance[msg.sender])() ) ){
        throw;
    }
    userBalance[msg.sender] = 0;
}
```

Bob uses the re-entrancy bug to call withdrawBalance two times, and withdraw more than its initial deposit to the contract.

● **Low-Risk:** Could be fixed, will not bring problems.

Too many digits

Literals with many digits are difficult to read and review.

```
uint256 private _totalSupply =  100000000 * 10**_decimals;
```

Recommendation

Use: Ether suffix, Time suffix, or The scientific notation

Exploit scenario

```
contract MyContract{
    uint 1_ether = 10000000000000000000;
}
```

While `1_ether` looks like `1 ether`, it is `10 ether`. As a result, it’s likely to be used incorrectly.

● **Low-Risk:** Could be fixed, will not bring problems.

No zero address validation for some functions

Detect missing zero address validation.

```
function setMarketingWalletAddress(address newAddress) external onlyOwner() {
    marketingWalletAddress = payable(newAddress);
}

function setNftWalletAddress(address newAddress) external onlyOwner() {
    nftWalletAddress = payable(newAddress);
}

function setAppWalletAddress(address newAddress) external onlyOwner() {
    appWalletAddress = payable(newAddress);
}
```

Recommendation

Check that the new address is not zero.

Exploit scenario

```
contract C {

  modifier onlyAdmin {
    if (msg.sender != owner) throw;
    _;
  }

  function updateOwner(address newOwner) onlyAdmin external {
    owner = newOwner;
  }
}
```

Bob calls `updateOwner` without specifying the `newOwner`, soBob loses ownership of the contract.

● **Low-Risk:** Could be fixed, will not bring problems.

Missing events arithmetic

Detect missing events for critical arithmetic parameters.

```
function setBuyTaxes(uint256 newLiquidityTax, uint256 newMarketingTax, uint256 newNftTax, uint256 ne
    require(newLiquidityTax.add(newMarketingTax).add(newNftTax).add(newAppTax) &lt;= 18, &quot;Tax e
    _buyLiquidityFee = newLiquidityTax;
    _buyMarketingFee = newMarketingTax;
    _buyNftFee = newNftTax;
    _buyAppFee = newAppTax;

    _totalTaxIfBuying = _buyLiquidityFee.add(_buyMarketingFee).add(_buyNftFee).add(_buyAppFee);
}

function setSelTaxes(uint256 newLiquidityTax, uint256 newMarketingTax, uint256 newNftTax, uint256 ne
    require(newLiquidityTax.add(newMarketingTax).add(newNftTax).add(newAppTax) &lt;= 18, &quot;Tax e
    _sellLiquidityFee = newLiquidityTax;
    _sellMarketingFee = newMarketingTax;
    _sellNftFee = newNftTax;
    _sellAppFee=newAppTax;

    _totalTaxIfSelling = _sellLiquidityFee.add(_sellMarketingFee).add(_sellNftFee).add(_sellAppFee);
}
```

Recommendation

Emit an event for critical parameter changes.

Exploit scenario

```
contract C {

  modifier onlyAdmin {
    if (msg.sender != owner) throw;
    _;
  }

  function updateOwner(address newOwner) onlyAdmin external {
    owner = newOwner;
  }
}
```

`updateOwner()` has no event, so it is difficult to track off-chain changes in the buy price.

Owner privileges

- 🟢 Owner cannot set fees higher than 25%
- 🟢 Owner cannot pause trading
- 🟢 Owner cannot change max transaction amount
- 🟡 Owner can exclude from fees

Extra notes by the team

Warning: Audit partner notified us that this project is likely to be related to Luck2Earn, which resulted in a scam. Please DYOR when investing in this project.

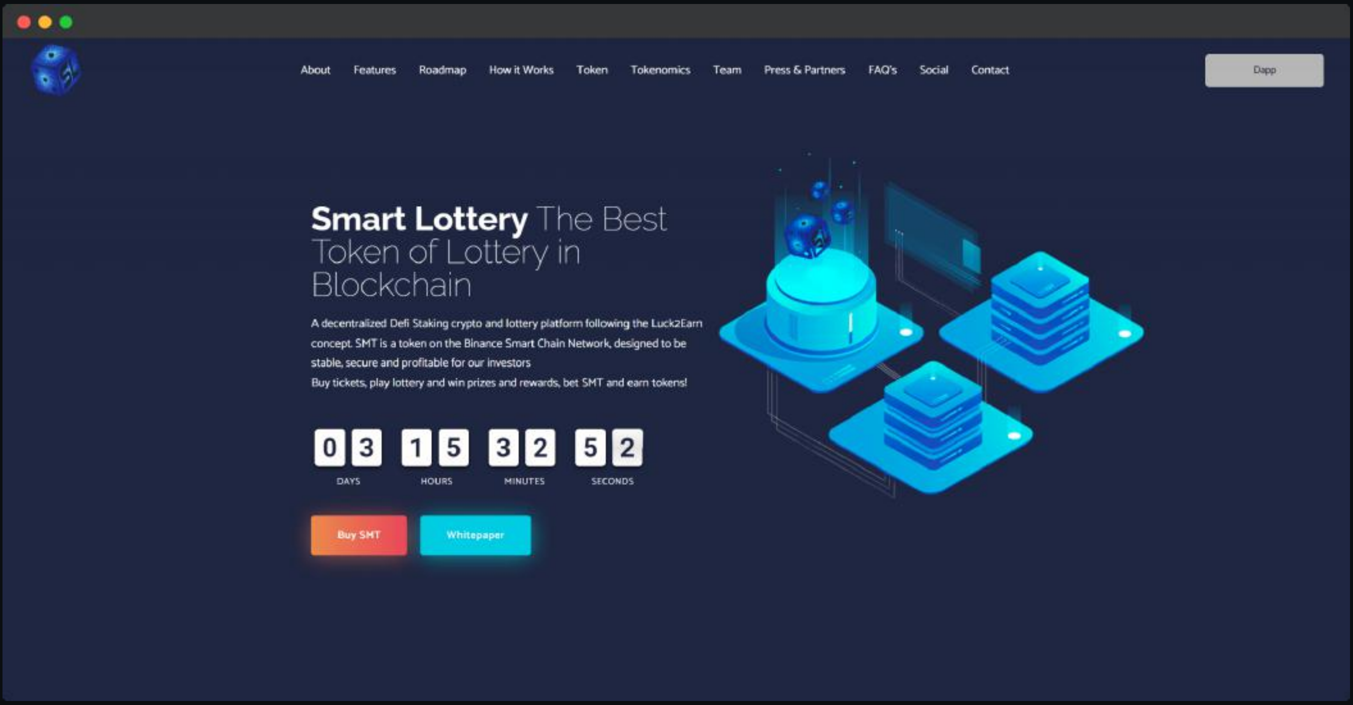
❌ KYC failed.

Contract Snapshot

```
contract SmartLottery is Context, IERC20, Ownable {

using SafeMath for uint256;
using Address for address;

string private _name = "SmartLottery";
string private _symbol = "SMT";
uint8 private _decimals = 18;


address payable public marketingWalletAddress = payable(0x79d67df82bE1469D9B6De9CaaC38217B64e8a319);
address payable public nftWalletAddress = payable(0x7aFCa110982714A8b58d91C25E9f268482B0Fa58);
address payable public appWalletAddress = payable(0xB6FA8Ae61133908f02E1E179C01cf6C838810233);
address public immutable deadAddress = 0x000000000000000000000000000000000000dEaD;

mapping (address =&gt; uint256) _balances;
mapping (address =&gt; mapping (address =&gt; uint256)) private _allowances;

mapping (address =&gt; bool) public isExcludedFromFee;
mapping (address =&gt; bool) public isMarketPair;

uint256 public _buyLiquidityFee = 0;
uint256 public _buyMarketingFee = 2;
uint256 public _buyNftFee = 0;
uint256 public _buyAppFee = 1;

uint256 public _sellLiquidityFee = 0;
uint256 public _sellMarketingFee = 3;
uint256 public _sellNftFee = 0;
uint256 public _sellAppFee = 1;

uint256 public _liquidityShare = 10;
uint256 public _marketingShare = 45;
uint256 public _nftShare = 15;
uint256 public _appShare = 30;

uint256 public _totalTaxIfBuying = 2;
```

Website Review

Coinsult checks the website completely manually and looks for visual, technical and textual errors. We also look at the security, speed and accessibility of the website. In short, a complete check to see if the website meets the current standard of the web development industry.



Smart Lottery The Best Token of Lottery in Blockchain

A decentralized Defi Staking crypto and lottery platform following the Luck2Earn concept. SMT is a token on the Binance Smart Chain Network, designed to be stable, secure and profitable for our investors
Buy tickets, play lottery and win prizes and rewards, bet SMT and earn tokens!

● Mobile Friendly

● Does not contain jQuery errors

● SSL Secured

● No major spelling errors

Project Overview

● Not KYC verified by Coinsult

Smart Lottery

Audited by Coinsult.net

Date: 12 July 2022

✓ Advanced Manual Smart Contract Audit