Coinsult

Advanced Manual Smart Contract Audit

Project: Crypto Sport Token
Website: https://cryptosporttoken.com

🟢 **Low-risk**	🟡 **Medium-risk**	🔴 **High-risk**
8 low-risk code issues found	1 medium-risk code issues found	0 high-risk code issues found

Contract address

0x7bE17051Ffa123EEfd6416495C185EC27bdf8d74 (relaunch)

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

Disclaimer

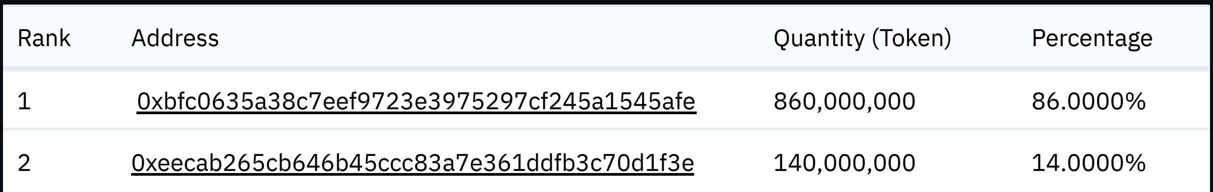
Coinsult is not responsible if a project turns out to be a scam, rug-pull or honeypot. We only provide a detailed analysis for your own research.

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	0xbfc0635a38c7eef9723e3975297cf245a1545afe	860,000,000	86.0000%
2	0xeecab265cb646b45ccc83a7e361ddfb3c70d1f3e	140,000,000	14.0000%

Source code

Coinsult was commissioned by Crypto Sport Token to perform an audit based on the following smart contract:

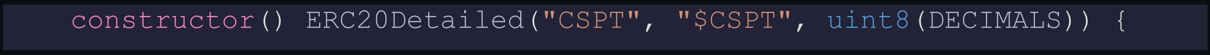
https://bscscan.com/address/0x7bE17051Ffa123EEfd6416495C185EC27bdf8d74#code

Manual Code Review

● Low-risk

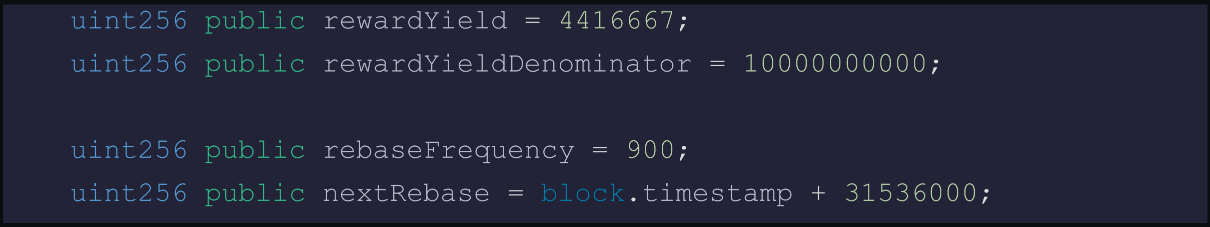
6 low-risk code issues found.
Could be fixed, will not bring problems.

- Contract symbol uses a non-alphanumeric characters ($) this can cause problems during CEX listings



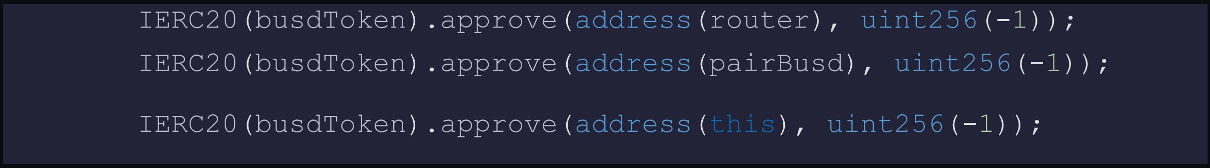
```
constructor() ERC20Detailed("CSPT", "$CSPT", uint8(DECIMALS)) {
```

- Literals with many digits are difficult to read and review.
  Recommendation: Use Ether suffix, Time suffix, or The scientific notation



```
uint256 public rewardYield = 4416667;
uint256 public rewardYieldDenominator = 10000000000;

uint256 public rebaseFrequency = 900;
uint256 public nextRebase = block.timestamp + 31536000;
```

- The return value of an external transfer/transferFrom call is not checked
  Use SafeERC20, or ensure that the transfer/transferFrom return value is checked.



```
IERC20(busdToken).approve(address(router), uint256(-1));
IERC20(busdToken).approve(address(pairBusd), uint256(-1));

IERC20(busdToken).approve(address(this), uint256(-1));
```

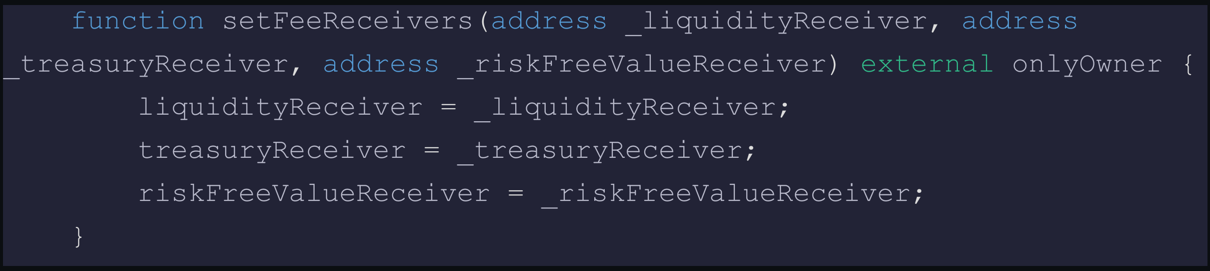
- Missing events for critical arithmetic parameters

  Emit an event for critical parameter changes.

```
    function setAutoRebase(bool _autoRebase) external onlyOwner {
        require(autoRebase != _autoRebase, "Not changed");
        autoRebase = _autoRebase;
    }

    function setRebaseFrequency(uint256 _rebaseFrequency) external
onlyOwner {
        require(_rebaseFrequency <= MAX_REBASE_FREQUENCY, "Too high");
        rebaseFrequency = _rebaseFrequency;
    }

    function setRewardYield(uint256 _rewardYield, uint256
_rewardYieldDenominator) external onlyOwner {
        rewardYield = _rewardYield;
        rewardYieldDenominator = _rewardYieldDenominator;
    }

    function setFeesOnNormalTransfers(bool _enabled) external onlyOwner
{
        require(feesOnNormalTransfers != _enabled, "Not changed");
        feesOnNormalTransfers = _enabled;
    }

    function setIsLiquidityInBnb(bool _value) external onlyOwner {
        require(isLiquidityInBnb != _value, "Not changed");
        isLiquidityInBnb = _value;
    }

    function setNextRebase(uint256 _nextRebase) external onlyOwner {
        nextRebase = _nextRebase;
    }
```

- No zero address validation

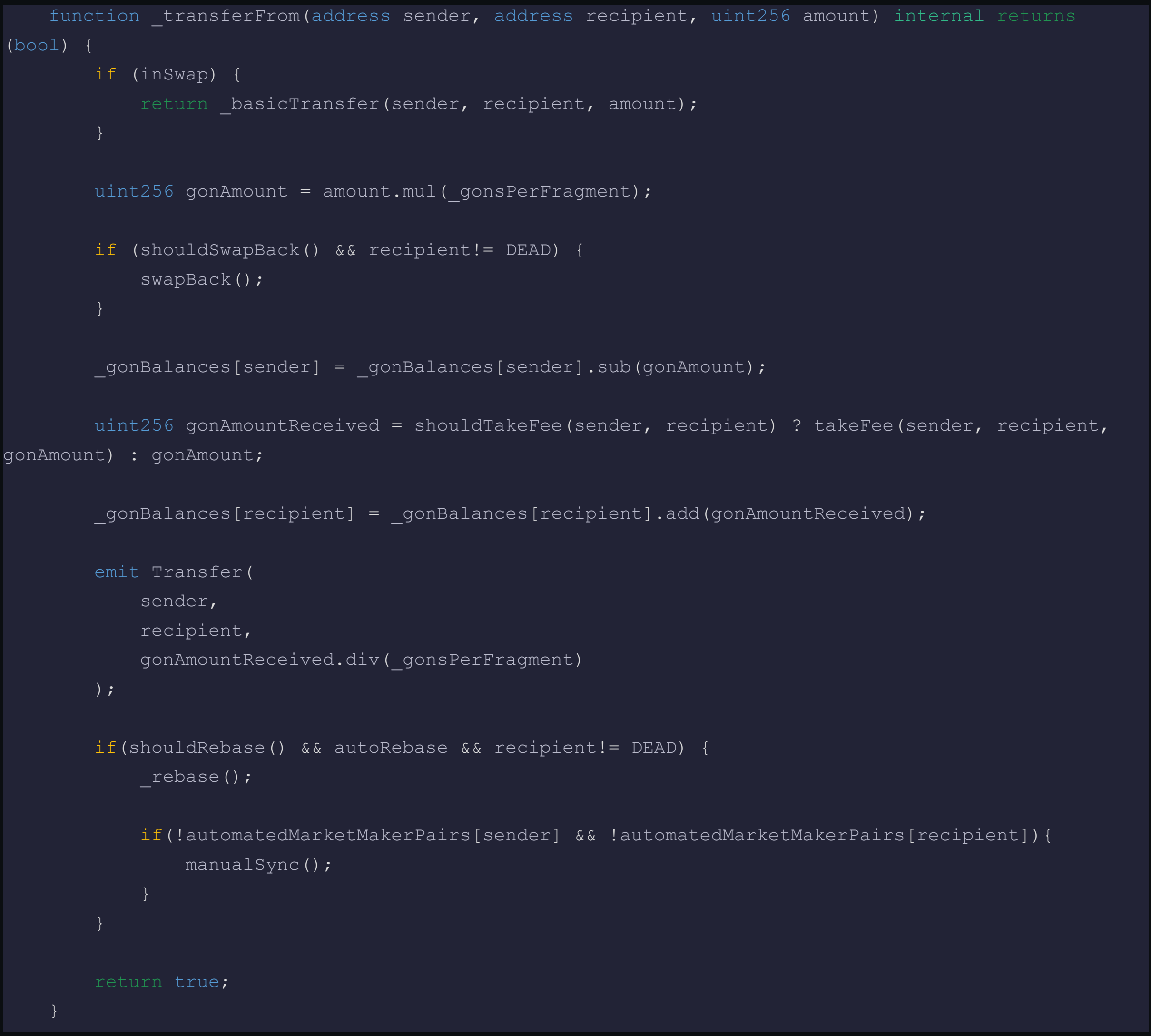
  Check that the new address is not the zero address.



```
    function setFeeReceivers(address _liquidityReceiver, address
_treasuryReceiver, address _riskFreeValueReceiver) external onlyOwner {
        liquidityReceiver = _liquidityReceiver;
        treasuryReceiver = _treasuryReceiver;
        riskFreeValueReceiver = _riskFreeValueReceiver;
    }
```

- Contract contains Reentrancy vulnerabilities: _transfer(address,address,uint256)

  This contract seems to have a dividend tracker but has unchecked transfer vulnerabilities



```
    function _transferFrom(address sender, address recipient, uint256 amount) internal returns
(bool) {
        if (inSwap) {
            return _basicTransfer(sender, recipient, amount);
        }

        uint256 gonAmount = amount.mul(_gonsPerFragment);

        if (shouldSwapBack() && recipient!= DEAD) {
            swapBack();
        }

        _gonBalances[sender] = _gonBalances[sender].sub(gonAmount);

        uint256 gonAmountReceived = shouldTakeFee(sender, recipient) ? takeFee(sender, recipient,
gonAmount) : gonAmount;

        _gonBalances[recipient] = _gonBalances[recipient].add(gonAmountReceived);

        emit Transfer(
            sender,
            recipient,
            gonAmountReceived.div(_gonsPerFragment)
        );

        if(shouldRebase() && autoRebase && recipient!= DEAD) {
            _rebase();

            if(!automatedMarketMakerPairs[sender] && !automatedMarketMakerPairs[recipient]){
                manualSync();
            }
        }

        return true;
    }
```

Medium-risk

1 medium-risk code issues found.
Should be fixed, could bring problems.

- Owner can pause trading

```
    function setSwapBackSettings(bool _enabled, uint256 _num, uint256
_denom) external onlyOwner {
        swapEnabled = _enabled;
        gonSwapThreshold = TOTAL_GONS.div(_denom).mul(_num);
    }
```

High-risk

0 high-risk code issues found
Must be fixed, and will bring problems.

Extra notes by the team

🟡 Owner can not change the fees

```
    uint256 public liquidityFee = 5;
    uint256 public treasuryFee = 3;
    uint256 public burnFee = 0;
    uint256 public buyFeeRFV = 5;
    uint256 public sellFeeTreasuryAdded = 5;
    uint256 public sellFeeRFVAdded = 6;
    uint256 public totalBuyFee =
liquidityFee.add(treasuryFee).add(buyFeeRFV).add(burnFee);
    uint256 public totalSellFee =
totalBuyFee.add(sellFeeTreasuryAdded).add(sellFeeRFVAdded);
    uint256 public feeDenominator = 100;

    uint256 targetLiquidity = 50;
    uint256 targetLiquidityDenominator = 100;
```

🟡 Owner can exclude from fees

🟡 The ownership of the contract isn't renounced

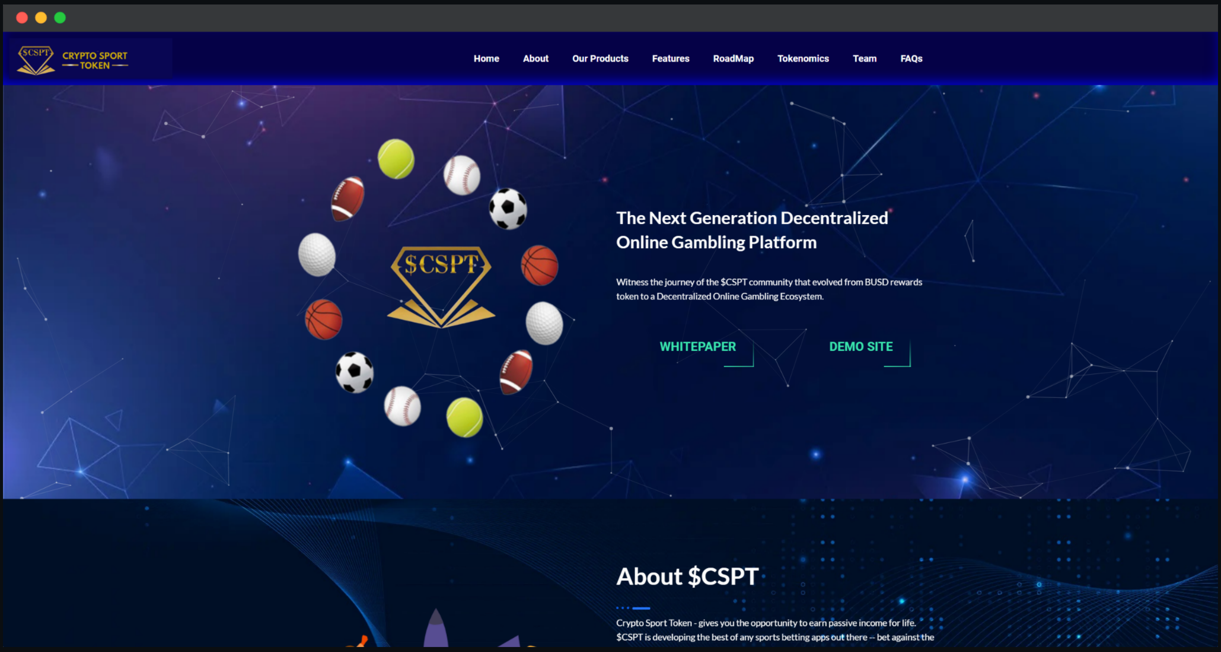
🔴 Owner can pause trading

```
    function setSwapBackSettings(bool _enabled, uint256 _num, uint256
_denom) external onlyOwner {
        swapEnabled = _enabled;
        gonSwapThreshold = TOTAL_GONS.div(_denom).mul(_num);
    }
```

Contract Snapshot

```
contract CSPT is ERC20Detailed, Ownable {
    using SafeMath for uint256;
    using SafeMathInt for int256;

    bool public swapEnabled = true;
    bool public autoRebase = true;
    bool public feesOnNormalTransfers = false;
    bool public isLiquidityInBnb = true;

    uint256 public rewardYield = 4416667;
    uint256 public rewardYieldDenominator = 10000000000;

    uint256 public rebaseFrequency = 900;
    uint256 public nextRebase = block.timestamp + 31536000;

    mapping(address => bool) _isFeeExempt;
    address[] public _markerPairs;
    mapping (address => bool) public automatedMarketMakerPairs;

    uint256 public constant MAX_FEE_RATE = 18;
    uint256 private constant MAX_REBASE_FREQUENCY = 900;
    uint256 private constant DECIMALS = 18;
    uint256 private constant MAX_UINT256 = ~uint256(0);
    uint256 private constant INITIAL_FRAGMENTS_SUPPLY = 10 * 10**8 *
10**DECIMALS;
    uint256 private constant TOTAL_GONS = MAX_UINT256 - (MAX_UINT256 %
INITIAL_FRAGMENTS_SUPPLY);
    uint256 private constant MAX_SUPPLY = ~uint128(0);

    address DEAD = 0x000000000000000000000000000000000000dEaD;
    address ZERO = 0x0000000000000000000000000000000000000000;

    address public liquidityReceiver =
0x8568702031101f3Af42c62d8b27a021Fa655E199;
    address public treasuryReceiver =
0xDa5F94Fc23d7aE3f7db0870dd2b964946eC7887e;
    address public riskFreeValueReceiver =
0x4663D88AA62EE3067C1B1d2996B551b8E53b455a;
    address public busdToken =
0xe9e7CEA3DedcA5984780Bafc599bD69ADd087D56;
```

Website Review



The Next Generation Decentralized Online Gambling Platform

Witness the journey of the $CSPT community that evolved from BUSD rewards token to a Decentralized Online Gambling Ecosystem.

About $CSPT

Crypto Sport Token - gives you the opportunity to earn passive income for life. $CSPT is developing the best of any sports betting apps out there -- bet against the

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.

- 🟢 Mobile Friendly
- 🟢 Contains no jQuery errors
- 🟢 SSL Secured
- 🟢 No major spelling errors

Loading speed: 78%

Rug-pull Review

Based on the available information analyzed by us, we come to the following conclusions:

🟡 Locked Liquidity (no liquidity yet)

🟢 No large unlocked wallets

🟢 Doxxed Team (KYC)

Honeypot Review

Based on the available information analyzed by us, we come to the following conclusions:

🟡 Ability to sell

Owner can pause trading

🔴 Owner is able to pause the contract

🟢 Correct router hard coded in the contract

0x10ED43C718714eb63d5aA57B78B54704E256024E

Note: Please check the disclaimer above and note, the audit makes no statements or warranties on business model, investment attractiveness or code sustainability. The report is provided for the only contract mentioned in the report and does not include any other potential contracts deployed by the project owner.