Coinsult

Advanced Manual Smart Contract Audit

Project: CyborgMetacat
Website: https://www.cyborgmetacat.com/

🟢 Low-risk	🟡 Medium-risk	🔴 High-risk
5 low-risk code issues found	1 medium-risk code issues found	0 high-risk code issues found
Acknowledged ✅	Acknowledged ✅	

Contract address
0x18bcc4feda55e2272f60384eae2ade6c2ffb45c8

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

Total Supply: 200,000,000
Total Holders: 3
Top 10 holders:

Rank	Address	Quantity (Token)	Percentage
1	0xf9c0bf861408bdb3a932b4b1cf3f26db945261b4	108,640,009.8	54.3200%
2	0xa1860c22c1457512e05c8642f60e1ffb7ec26156	51,359,990.2	25.6800%
3	Null Address: 0x000...dEaD	40,000,000	20.0000%

The top 100 holders collectively own 100.00% (200,000,000.00 Tokens) of CyborgMetacat

Acknowledged: The 3 addresses shown are for the Liquidity pool, private/public presale, and burn address. No dev token.

Note: This is a snapshot of when the audit was performed.

Source code

Coinsult was commissioned by CyborgMetacat to perform an audit based on the following smart contract:

https://bscscan.com/address/0x18bcc4feda55e2272f60384eae2ade6c2ffb45c8#code

Manual Code Review

🟢 Low-risk

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

- Weak PRNG, do not use block.timestamp as a source of randomness as this can be manipulated by miners.

```
uint256 _bBSLimit = _bBSLimitMin +
uint256(keccak256(abi.encodePacked(block.timestamp, block.difficulty)))
% (_bBSLimitMax - _bBSLimitMin + 1);
```

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

```
    string private _symbol = "$CYCAT";
```

- Contract has relative high sell fees and low buy fees
    - Acknowledged: This is to prevent dumps after presale.

```
    uint256 public _buyTaxFee = 1;
    uint256 public _buyLiquidityFee = 1;

    uint256 public _sellTaxFee = 15;
    uint256 public _sellLiquidityFee = 7;
```

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

  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

```
function _transfer(
        address from,
        address to,
        uint256 amount
    ) private {
        require(from != address(0), "ERC20: transfer from the zero address");
        require(to != address(0), "ERC20: transfer to the zero address");
        require(amount > 0, "Transfer amount must be greater than zero");
        if(from != owner() && to != owner()) {
            require(amount <= _maxTxAmount, "Transfer amount exceeds the maxTxAmount.");
        }

        uint256 contractTokenBalance = balanceOf(address(this));
        bool overMinimumTokenBalance = contractTokenBalance >= minimumTokensBeforeSwap;

        if (to == uniswapV2Pair && balanceOf(uniswapV2Pair) > 0) {
            SellHistories memory sellHistory;
            sellHistory.time = block.timestamp;
            sellHistory.bnbAmount = _getSellBnBAmount(amount);

            _sellHistories.push(sellHistory);
        }

        // Sell tokens for ETH
        if (!inSwapAndLiquify && swapAndLiquifyEnabled && balanceOf(uniswapV2Pair) > 0) {
            if (to == uniswapV2Pair) {
                if (overMinimumTokenBalance && _startTimeForSwap + _intervalMinutesForSwap
<= block.timestamp) {
                    _startTimeForSwap = block.timestamp;
                    contractTokenBalance = minimumTokensBeforeSwap;
                    swapTokens(contractTokenBalance);
                }

                if (buyBackEnabled) {

                    uint256 balance = address(this).balance;

                    uint256 _bBSLimitMax = buyBackSellLimit;

                    if (_isAutoBuyBack) {

                        uint256 sumBnbAmount = 0;
                        uint256 startTime = block.timestamp - _buyBackTimeInterval;
```

```
                        uint256 cnt = 0;

                        for (uint i = 0; i < _sellHistories.length; i ++) {

                            if (_sellHistories[i].time >= startTime) {
                                sumBnbAmount =
sumBnbAmount.add(_sellHistories[i].bnbAmount);
                                cnt = cnt + 1;
                            }
                        }

                        if (cnt > 0 && _buyBackDivisor > 0) {
                            _bBSLimitMax = sumBnbAmount.div(cnt).div(_buyBackDivisor);
                        }

                        _removeOldSellHistories();
                    }

                    uint256 _bBSLimitMin = _bBSLimitMax.mul(_buyBackRangeRate).div(100);

                    uint256 _bBSLimit = _bBSLimitMin +
uint256(keccak256(abi.encodePacked(block.timestamp, block.difficulty))) % (_bBSLimitMax -
_bBSLimitMin + 1);

                    if (balance > _bBSLimit) {
                        buyBackTokens(_bBSLimit);
                    }
                }
            }

        }

        bool takeFee = true;

        // If any account belongs to _isExcludedFromFee account then remove the fee
        if(_isExcludedFromFee[from] || _isExcludedFromFee[to]){
            takeFee = false;
        }
        else{
            // Buy
            if(from == uniswapV2Pair){
                removeAllFee();
                _taxFee = _buyTaxFee;
                _liquidityFee = _buyLiquidityFee;
            }
            // Sell
            if(to == uniswapV2Pair){
                removeAllFee();
                _taxFee = _sellTaxFee;
                _liquidityFee = _sellLiquidityFee;
            }

            // If send account has a special fee
```

```
            if(_addressFees[from].enable){
                removeAllFee();
                _taxFee = _addressFees[from]._taxFee;
                _liquidityFee = _addressFees[from]._liquidityFee;

                // Sell
                if(to == uniswapV2Pair){
                    _taxFee = _addressFees[from]._sellTaxFee;
                    _liquidityFee = _addressFees[from]._sellLiquidityFee;
                }
            }
            else{
                // If buy account has a special fee
                if(_addressFees[to].enable){
                    //buy
                    removeAllFee();
                    if(from == uniswapV2Pair){
                        _taxFee = _addressFees[to]._buyTaxFee;
                        _liquidityFee = _addressFees[to]._buyLiquidityFee;
                    }
                }
            }
        }

        _tokenTransfer(from,to,amount,takeFee);
    }
```

- To many digits (Use: Ether suffix, Time suffix, or The scientific notation)

```
    _maxTxAmount = 1000000000 * 10**6 * 10**9;
```

🟡 Medium-risk

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

- Owner can change router address

```
function changeRouterVersion(address _router) public onlyOwner
returns(address _pair) {
        IUniswapV2Router02 _uniswapV2Router =
IUniswapV2Router02(_router);

        _pair =
IUniswapV2Factory(_uniswapV2Router.factory()).getPair(address(this),
_uniswapV2Router.WETH());
        if(_pair == address(0)){
```

```
            // Pair doesn't exist
            _pair = IUniswapV2Factory(_uniswapV2Router.factory())
            .createPair(address(this), _uniswapV2Router.WETH());
        }
        uniswapV2Pair = _pair;

        // Set the router of the contract variables
        uniswapV2Router = _uniswapV2Router;
    }
```

🔴 High-risk

0 high-risk code issues found

Must be fixed, and will bring problems.

Extra notes by the team

There are unnecessarily many functions which will result in an unstructured code. For example there are 6 different functions to change the fees. These could all be in one function.

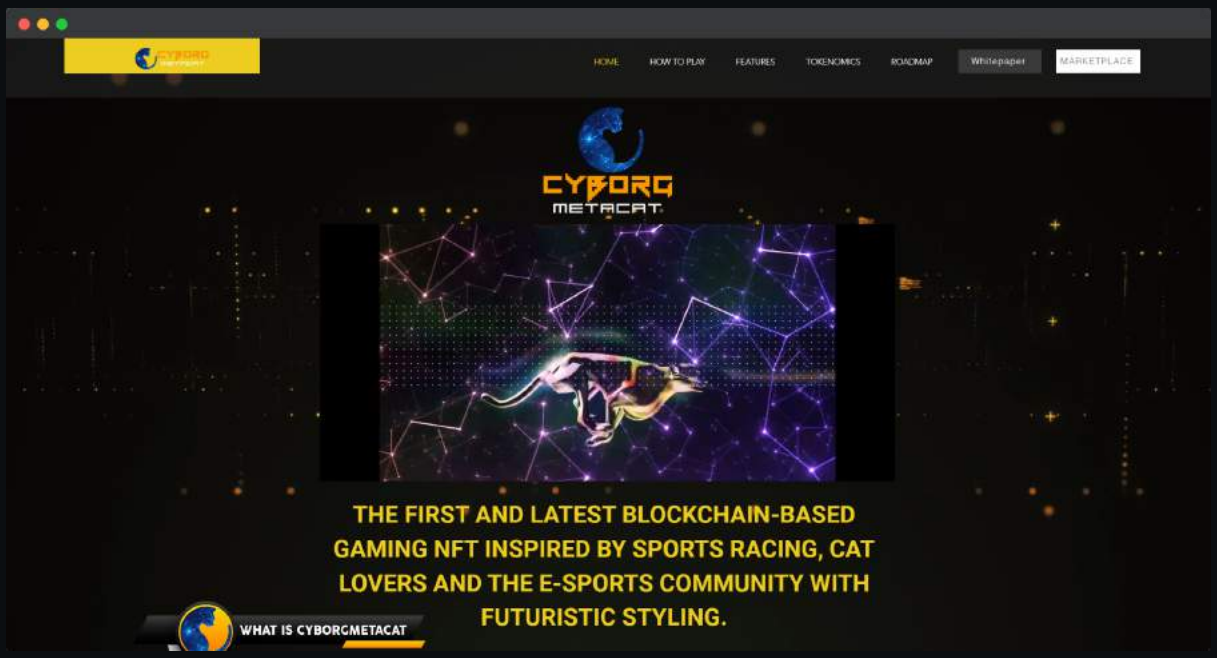
Notes:

- Owner entitled to change the transaction fees up to 100%
- Owner can set max transaction amount
- Owner can whitelist addresses from fee
- The ownership of the contract isn’t renounced
- Testnet router address also in code for mainnet useless as it will not be used

Contract Snapshot

```
contract $CYCAT is Context, IERC20, Ownable {
    using SafeMath for uint256;
    using Address for address;

    address payable public marketingAddress =
payable(0xa1860C22C1457512E05C8642F60E1fFb7eC26156); // Marketing
Address
    address public immutable deadAddress =
0x000000000000000000000000000000000000dEaD;
    mapping (address => uint256) private _rOwned;
    mapping (address => uint256) private _tOwned;
    mapping (address => mapping (address => uint256)) private
_allowances;

    mapping (address => bool) private _isExcludedFromFee;

    mapping (address => bool) private _isExcluded;
    address[] private _excluded;

    uint256 private constant MAX = ~uint256(0);
    uint256 private _tTotal = 200 * 10**6 * 10**9;
    uint256 private _rTotal = (MAX - (MAX % _tTotal));
    uint256 private _tFeeTotal;

    string private _name = "CyborgMetacat";
    string private _symbol = "$CYCAT";
    uint8 private _decimals = 9;
```

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.

- Mobile Friendly (However some scrolling issues might occur)
- No jQuery errors
- SSL Secured
- Appropriate spelling

Note: Try to avoid using images as text. By using images you avoid search engines to index your content and you avoid accessibility on all devices.

Loading speed: 91%

Rug-pull Review

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

- Locked Liquidity ✅
- Large unlocked wallets ⚠ (26%)
- No Doxxed Team ❌

Honeypot Review

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

- Ability to sell ⚠
  (Fees can be set higher than 25% by owner)
- Owner unable to prevent selling ✅
  (But, fees can be set higher than 25% by owner)
- Accurate liquidity pair ✅
  (At the moment of the audit, but this can be changed using a function stated under medium-risk)

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.