Coinsult

Advanced Manual Smart Contract Audit

Project: Crypto Fan Token

Website: https://criptofan.com.br

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

Contract Address

0x5BeB1898ae55Fc1e65F65f61DB101Ab8684DF05B

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	0x7210f2c0c34a80711b67745b7e5d22383410dd0e	1,000,000,000	100.0000%

Source Code

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

https://bscscan.com/address/0x5BeB1898ae55Fc1e65F65f61DB101Ab8684DF05B#code

Manual Code Review

In this audit report we will highlight all these issues:

🟢 Low-Risk

2 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.

No constraints on lock time

```
//Locks the contract for owner for the amount of time provided
function lock(uint256 time) public virtual onlyOwner {
    _previousOwner = _owner;
    _owner = address(0);
    _lockTime = now + time;
    emit OwnershipTransferred(_owner, address(0));
}
```

Recommendation

It is recommended to put a require statement to prevent the lock time from exceeding a specific amount. For example, when you try to lock for 1 year, but you make it 10, you lose ownership for 10 years.

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

Costly operations inside a loop

Costly operations inside a loop might waste gas, so optimizations are justified.

```
function includeInReward(address account) external onlyOwner() {
    require(_isExcluded[account], "Account is already excluded");
    for (uint256 i = 0; i &lt; _excluded.length; i++) {
        if (_excluded[i] == account) {
            _excluded[i] = _excluded[_excluded.length - 1];
            _tOwned[account] = 0;
            _isExcluded[account] = false;
            _excluded.pop();
            break;
        }
    }
}
```

Recommendation

Use a local variable to hold the loop computation result.

Exploit scenario

```
contract CostlyOperationsInLoop{

    function bad() external{
        for (uint i=0; i < loop_count; i++){
            state_variable++;
        }
    }

    function good() external{
      uint local_variable = state_variable;
      for (uint i=0; i < loop_count; i++){
        local_variable++;
      }
      state_variable = local_variable;
    }
}
```

Incrementing `state_variable` in a loop incurs a lot of gas because of expensive `SSTORE`s, which might lead to an `out-of-gas`.

Owner privileges

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

⚠️ Owner can lock the contract, so that he loses ownership of the contract for a specific ‘time’

⚠️ Owner can set max wallet percentage

Extra notes by the team

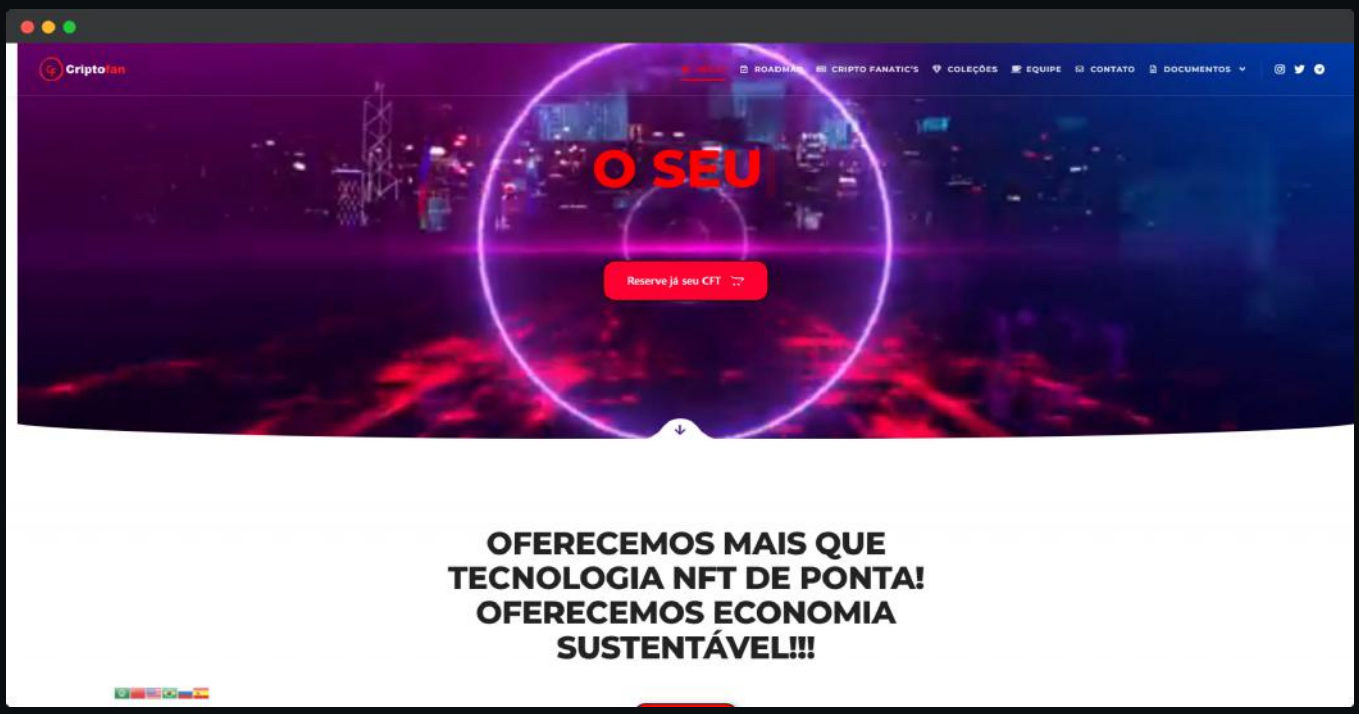
No notes

Contract Snapshot

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

mapping (address =&gt; uint256) private _rOwned;
mapping (address =&gt; uint256) private _tOwned;
mapping (address =&gt; mapping (address =&gt; uint256)) private _allowances;

mapping (address =&gt; bool) private _isExcludedFromFee;

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

uint256 private constant MAX = ~uint256(0);
uint256 private _tTotal = 1 * 10**9 * 10**9; // 1 billion
uint256 private _rTotal = (MAX - (MAX % _tTotal));
uint256 private _tFeeTotal;

string private _name = "CryptoFanToken";
string private _symbol = "CFT";
uint8 private _decimals = 9;

uint256 public _burnFee = 0;
uint256 private _previousBurnFee = _burnFee;

uint256 public _marketingFee = 6;
uint256 private _previousMarketingFee = _marketingFee;

uint256 public _taxFee = 1;
uint256 private _previousTaxFee = _taxFee;

uint256 public _liquidityFee = 2;
uint256 private _previousLiquidityFee = _liquidityFee;

address public marketingWallet = 0x7210F2c0C34A80711B67745b7e5d22383410dd0e;

IUniswapV2Router02 public immutable uniswapV2Router;
```

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
- Does not contain jQuery errors
- SSL Secured
- No major spelling errors

Project Overview

Not KYC verified by Coinsult