Coinsult

Advanced Manual Smart Contract Audit

Project: TGG
Website: -

🟢 **Low-risk**
6 low-risk code issues found

🟡 **Medium-risk**
0 medium-risk code issues found

🔴 **High-risk**
0 high-risk code issues found

Contract address
0xA810d40Ca60A3722287664a0147413829CD3A6bB

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

Disclaimer

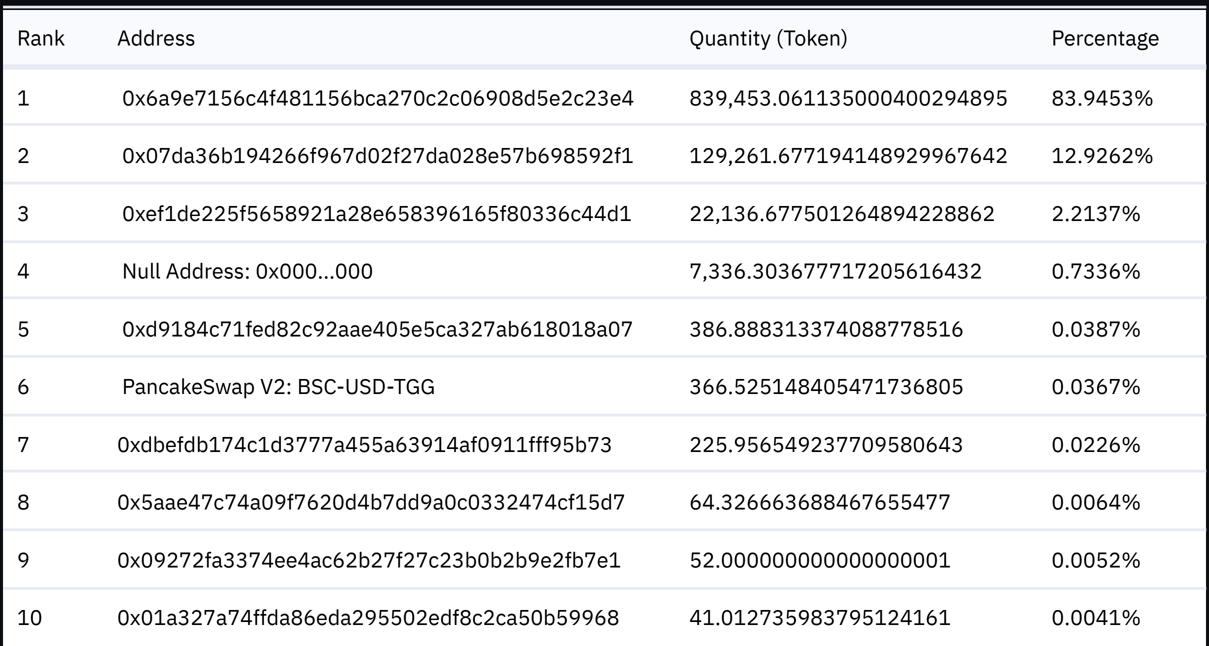
Coinsult is not responsible if a project turns out to be a scam, rug-pull or honeypot. We only provide a detailed analysis for your own research.

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	0x6a9e7156c4f481156bca270c2c06908d5e2c23e4	839,453.061135000400294895	83.9453%
2	0x07da36b194266f967d02f27da028e57b698592f1	129,261.677194148929967642	12.9262%
3	0xef1de225f5658921a28e658396165f80336c44d1	22,136.677501264894228862	2.2137%
4	Null Address: 0x000...000	7,336.303677717205616432	0.7336%
5	0xd9184c71fed82c92aae405e5ca327ab618018a07	386.888313374088778516	0.0387%
6	PancakeSwap V2: BSC-USD-TGG	366.525148405471736805	0.0367%
7	0xdbefdb174c1d3777a455a63914af0911fff95b73	225.956549237709580643	0.0226%
8	0x5aae47c74a09f7620d4b7dd9a0c0332474cf15d7	64.326663688467655477	0.0064%
9	0x09272fa3374ee4ac62b27f27c23b0b2b9e2fb7e1	52.000000000000000001	0.0052%
10	0x01a327a74ffda86eda295502edf8c2ca50b59968	41.012735983795124161	0.0041%

Source code

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

https://bscscan.com/address/0xA810d40Ca60A3722287664a0147413829CD3A6bB#code

Note: This project uses a proxy contract. While we do check the full contract for vulnerabilities at the time of the audit, we can not ensure the correctness of the proxied contract.

Proxy currently points at:
https://bscscan.com/address/0xb18d6b07d81e1af1c2c5017b8df6420f67f527bc#code

And we audited: File 9 of 11 : TGG_Token.sol

Manual Code Review

● Low-risk

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

- 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 sender,
        address recipient,
        uint256 amount
    ) internal override {
        require(contractStatus);
        require(sender != address(0), "ERC20: transfer from the zero
address");
        require(!black[msg.sender] && !black[sender] &&
!black[recipient], 'black');
        uint256 senderBalance = tokenHoldersMap.values[sender];
        require(senderBalance >= amount, "ERC20: transfer amount
exceeds balance");
        set(sender, senderBalance - amount);
        uint recipientBalance = tokenHoldersMap.values[recipient];
        set(recipient, recipientBalance + amount);
        if (balanceOf(sender) == 0) {
            remove(sender);
        }
        uint tempDebt = withdrawnDividends[sender] * amount /
senderBalance;
        withdrawnDividends[recipient] += tempDebt;
        withdrawnDividends[sender] -= tempDebt;
        emit Transfer(sender, recipient, amount);


    }
```

- Consider using .add and .sub for addition and subtraction

```
        withdrawnDividends[recipient] += tempDebt;
        withdrawnDividends[sender] -= tempDebt;
```

- Block.timestamp can be manipulated by miners.
  Avoid relying on block.timestamp.

  More information:
  https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp

```
        if (amount > 0 && balanceOf(account) >= limitBalance) {
            lastClaimTimes[account] = block.timestamp;
            emit Claim(account, amount, automatic);
            return true;
        }
```

- Commented function should be removed
  This function is fully commented out and should therefore be removed from the code.

```
    //    function AirDrop(address[] memory list1, uint[] memory list2)
external onlyOwner{
    //        for(uint i = 0; i < list1.length; i ++){
    //            tokenHoldersMap.values[msg.sender] -= list2[i];
    //            tokenHoldersMap.values[list1[i]] += list2[i];
    //           emit Transfer(msg.sender, list1[i], list2[i]);
    //        }
    //    }
```

- Potential spelling mistake in fristBuyMode
  Spelling errors are a common reason for mistakes within smart contracts.

```
    function setFristBuy(bool b) external onlyOwner {
        fristBuyMode = b;
    }
```

- Missing zero address validation

  Check that the new address is not zero.

```
    function setRefer(address addr) external onlyOwner {
        refer = Refer(addr);
    }

    function setWallet(address com_, address market_, address airDrop_,
address fund_) external onlyOwner {
        com = com_;
        market = market_;
        airDrop = airDrop_;
        fund = fund_;
    }
```

🟡 Medium-risk

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

🔴 High-risk

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

Extra notes by the team

- Owner can blacklist addresses

```
function setBlack(address addr, bool b) external onlyOwner {
    black[addr] = b;
}
```

- Owner can pause trading

```
function setContractStatus(bool b) external onlyOwner {
    contractStatus = b;
}
```

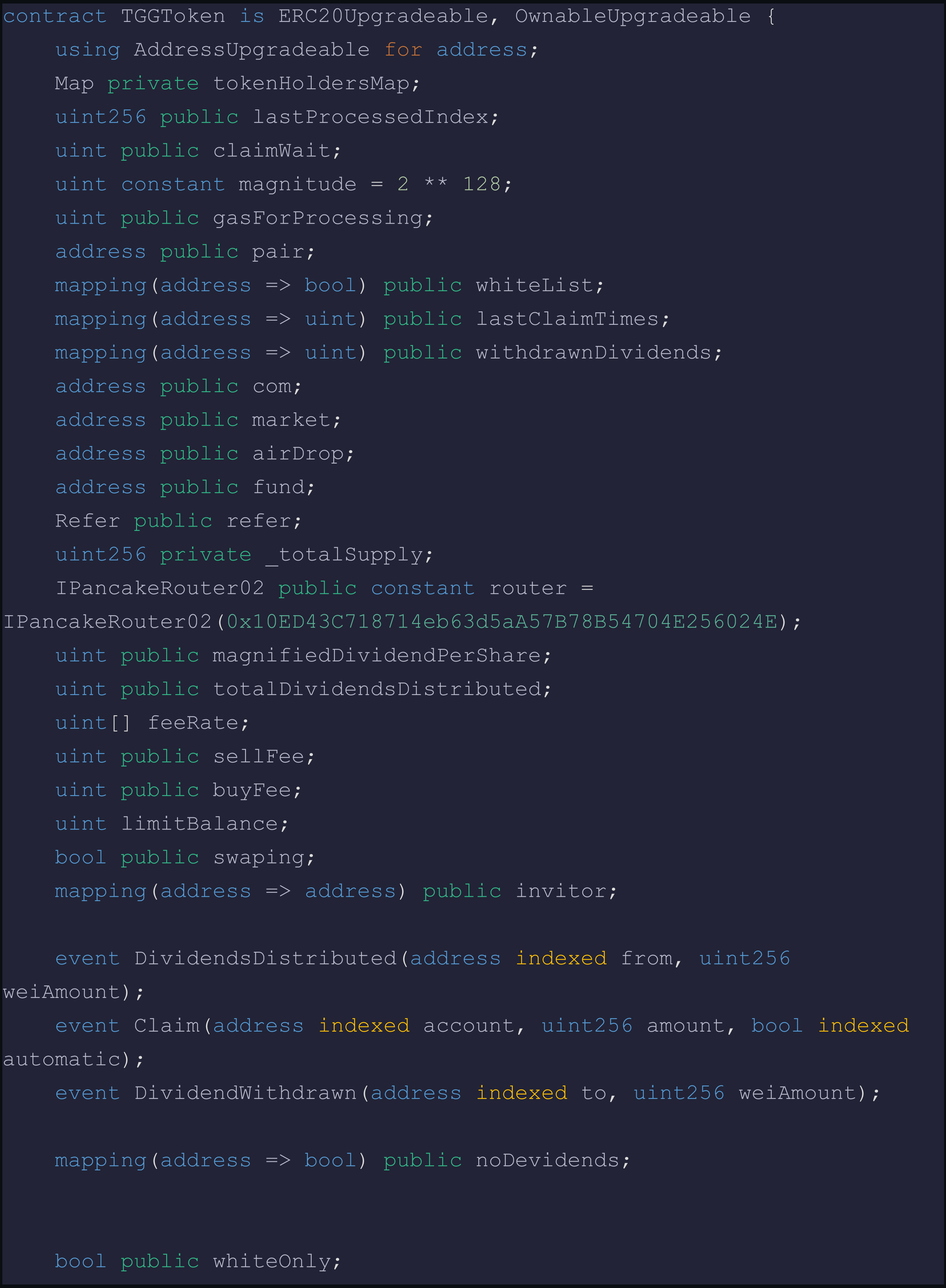
- Owner can change fee without a limit

```
function setFee(uint buy, uint sell) external onlyOwner {
    buyFee = buy;
    sellFee = sell;
}
```

- Owner can exclude from dividends

- Owner can set contract to whitelisted mode only

Contract Snapshot



```
contract TGGToken is ERC20Upgradeable, OwnableUpgradeable {
    using AddressUpgradeable for address;
    Map private tokenHoldersMap;
    uint256 public lastProcessedIndex;
    uint public claimWait;
    uint constant magnitude = 2 ** 128;
    uint public gasForProcessing;
    address public pair;
    mapping(address => bool) public whiteList;
    mapping(address => uint) public lastClaimTimes;
    mapping(address => uint) public withdrawnDividends;
    address public com;
    address public market;
    address public airDrop;
    address public fund;
    Refer public refer;
    uint256 private _totalSupply;
    IPancakeRouter02 public constant router =
IPancakeRouter02(0x10ED43C718714eb63d5aA57B78B54704E256024E);
    uint public magnifiedDividendPerShare;
    uint public totalDividendsDistributed;
    uint[] feeRate;
    uint public sellFee;
    uint public buyFee;
    uint limitBalance;
    bool public swaping;
    mapping(address => address) public invitor;

    event DividendsDistributed(address indexed from, uint256
weiAmount);
    event Claim(address indexed account, uint256 amount, bool indexed
automatic);
    event DividendWithdrawn(address indexed to, uint256 weiAmount);

    mapping(address => bool) public noDevidends;


    bool public whiteOnly;
```