Coinsult

Advanced Manual Smart Contract Audit

Project: Fishgun Token

Website: https://fishguncrypto.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

0x3D45BDCa2Ce083fA1002d557e3286dBd367923DC

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

-

Source Code

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

https://bscscan.com/address/0x3D45BDCa2Ce083fA1002d557e3286dBd367923DC#code

Safu Contract

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.

Inaccurate require statement

```
function updateBuyFees(uint256 _marketingFeeOnBuy) external onlyOwner {
    require(
        _marketingFeeOnBuy &amp;lt;= 3,
        &amp;quot;Fees must be less than 3%&amp;quot;
    );
    _totalFeesOnBuy = _marketingFeeOnBuy;
    emit UpdateBuyFees(_totalFeesOnBuy);
}

function updateSellFees(uint256 _marketingFeeOnSell) external onlyOwner {
    require(
        _marketingFeeOnSell &amp;lt;= 3,
        &amp;quot;Fees must be less than 3%&amp;quot;
    );
    _totalFeesOnSell = _marketingFeeOnSell;
    emit UpdateSellFees(_totalFeesOnSell);
}
```

Recommendation

Require statement states the fee must be less than 3% while it can also be 3%.

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

No zero address validation for some functions

Detect missing zero address validation.

```
function changeMarketingWallet(address _marketingWallet) external onlyOwner {
    require(_marketingWallet != marketingWallet, "Marketing wallet is already that address");
    require(!isContract(_marketingWallet), "Marketing wallet cannot be a contract");
    marketingWallet = _marketingWallet;
    emit MarketingWalletChanged(marketingWallet);
}
```

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.

Unchecked transfer

The return value of an external transfer/transferFrom call is not checked.

```
function claimStuckTokens(address token) external onlyOwner {
    require(token != address(this), "Owner cannot claim native tokens");
    if (token == address(0x0)) {
        payable(msg.sender).transfer(address(this).balance);
        return;
    }
    IERC20 ERC20token = IERC20(token);
    uint256 balance = ERC20token.balanceOf(address(this));
    ERC20token.transfer(msg.sender, balance);
}
```

Recommendation

Use `SafeERC20`, or ensure that the transfer/transferFrom return value is checked.

Exploit scenario

```
contract Token {
    function transferFrom(address _from, address _to, uint256 _value) public returns (bool success);
}
contract MyBank{
    mapping(address => uint) balances;
    Token token;
    function deposit(uint amount) public{
        token.transferFrom(msg.sender, address(this), amount);
        balances[msg.sender] += amount;
    }
}
```

Several tokens do not revert in case of failure and return false. If one of these tokens is used in `MyBank`, `deposit` will not revert if the transfer fails, and an attacker can call `deposit` for free..

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

Missing events arithmetic

Detect missing events for critical arithmetic parameters.

```
function setSwapTokensAtAmount(uint256 newAmount) external onlyOwner {
    require(
        newAmount &gt; totalSupply() / 1_000_000,
        "New Amount must more than 0.0001% of total supply"
    );
    swapTokensAtAmount = newAmount;
}
```

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

⚠️ Owner is in this case the ‘operator’ which can be changed by the operator role.

⚠️ Owner can update the UniswapV2Router

Extra notes by the team

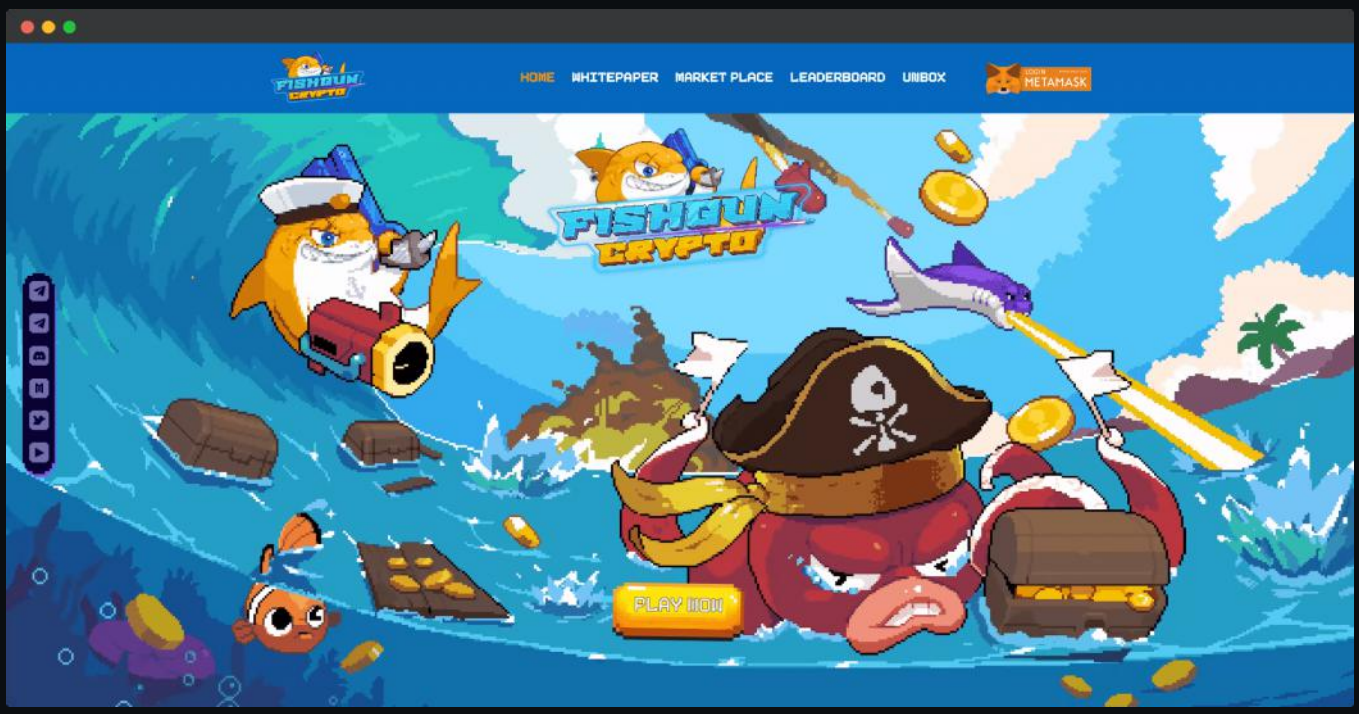
No notes

Contract Snapshot

```
contract FishgunToken is ERC20, Ownable {
using Address for address payable;

uint256 public _totalFeesOnBuy = 1;
uint256 public _totalFeesOnSell = 1;

address public  marketingWallet = 0x0f32c3EC010175FE26802E61eB43c3cAdB8B6820;

bool    public  walletToWalletTransferWithoutFee = false;

IUniswapV2Router02 public uniswapV2Router;
address public  uniswapV2Pair;

address private DEAD = 0x000000000000000000000000000000000000dEaD;

uint256 public swapTokensAtAmount;
bool    public swapWithLimit;
bool    private swapping;
bool    public swapEnabled = true;

mapping (address =&gt; bool) private _isExcludedFromFees;
mapping (address =&gt; bool) public automatedMarketMakerPairs;

event ExcludeFromFees(address indexed account);
event UpdateBuyFees(uint256 marketingFeeOnBuy);
event UpdateSellFees(uint256 marketingFeeOnSell);
event MarketingWalletChanged(address marketingWallet);
event SetAutomatedMarketMakerPair(address indexed pair, bool indexed value);
event SendMarketing(uint256 bnbSend);
event UpdateUniswapV2Router(address indexed newAddress, address indexed oldAddress);

constructor(
    address newOwner
)ERC20("Fishgun Token",  "FGC") {
    transferOwnership(newOwner);
```

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

Fishgun Token

Audited by Coinsult.net

Date: 27 July 2022

✔ Advanced Manual Smart Contract Audit