Coinsult

Advanced Manual Smart Contract Audit

Project: Treasure Hunt Token

Website: https://treasurehunttoken.com

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

Contract Address

0x8f3D0850a43C5323D0dF6B7FF951749678bFf6cd

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	0x77bb1363012fa74fc36e78a829c90ee29baa7226	237,637,000	79.2123%
2	0x0c07136a608ffb4fb6a7d826accd5f4211c77a30	62,363,000	20.7877%

Source Code

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

https://bscscan.com/address/0x8f3D0850a43C5323D0dF6B7FF951749678bFf6cd#code

Manual Code Review

In this audit report we will highlight all these issues:

🟢 Low-Risk

5 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.

Contract contains Reentrancy vulnerabilities

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 _tokenTransfer(address sender, address recipient, uint256 amount, bool isBuy, bool isSell)
    _beforeTokenTransfer(sender, recipient, amount);

    unchecked {
        _balances[sender] = balanceOf(sender) - amount;
    }

    if (isSell) {
        _balances[recipient] += deductSellFees(amount, sender);
    } else if (isBuy) {
        _balances[recipient] += deductBuyFees(amount, sender);
    } else {
        _balances[recipient] += amount;

        emit Transfer(sender, recipient, amount);
    }

    _afterTokenTransfer(sender, recipient, amount);
}

function deductBuyFees(uint256 amount, address sender) private returns(uint256) {
    uint256 totalFees = percent(amount, buyFeeRates.treasureNFee)
```

Recommendation

Apply the check-effects-interactions pattern.

Exploit scenario

```
function withdrawBalance(){
    // send userBalance[msg.sender] Ether to msg.sender
    // if mgs.sender is a contract, it will call its fallback function
    if( ! (msg.sender.call.value(userBalance[msg.sender])() ) ){
        throw;
    }
    userBalance[msg.sender] = 0;
}
```

Bob uses the re-entrancy bug to call withdrawBalance two times, and withdraw more than its initial deposit to the contract.

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

Too many digits

Literals with many digits are difficult to read and review.

```
uint256 public swapTokensAtAmount = 100_000 * 10**18;
```

Recommendation

Use: Ether suffix, Time suffix, or The scientific notation

Exploit scenario

```
contract MyContract{
    uint 1_ether = 10000000000000000000;
}
```

While `1_ether` looks like `1 ether`, it is `10 ether`. As a result, it’s likely to be used incorrectly.

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

No zero address validation for some functions

Detect missing zero address validation.

```
function setAddresses(address _treasureFeeAddr, address _marketingFeeAddr, address _operationFeeAddr
onlyOwner {
        treasureFeeAddr = _treasureFeeAddr;
        marketingFeeAddr = _marketingFeeAddr;
        operationFeeAddr = _operationFeeAddr;
}
```

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.

Missing events arithmetic

Detect missing events for critical arithmetic parameters.

```
function setFeeRates(feeRateStruct memory _buyFeeRates, feeRateStruct memory _sellFeeRates) external
    uint256 buyFees = _buyFeeRates.treasureNFee
            .add(_buyFeeRates.liquidityFee)
            .add(_buyFeeRates.treasureFee)
            .add(_buyFeeRates.marketingFee)
            .add(_buyFeeRates.operationFee);

    uint256 sellFees = _sellFeeRates.treasureNFee
            .add(_sellFeeRates.liquidityFee)
            .add(_sellFeeRates.treasureFee)
            .add(_sellFeeRates.marketingFee)
            .add(_sellFeeRates.operationFee);

    require(buyFees &lt;= 1200, &quot;buy fees above limt&quot;);
    require(sellFees &lt;= 1200, &quot;sell fees above limt&quot;);

    buyFeeRates = _buyFeeRates;
    sellFeeRates = _sellFeeRates;
}
```

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.

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

Boolean equality

Detects the comparison to boolean constants.

```
require(blacklist[sender] == false &amp;&amp; blacklist[recipient] == false, "You are blacklisted");
```

Recommendation

Remove the equality to the boolean constant.

Exploit scenario

```
contract A {
    function f(bool x) public {
        // ...
        if (x == true) { // bad!
           // ...
        }
        // ...
    }
}
```

Boolean constants can be used directly and do not need to be compare to `true` or `false`.

Owner privileges

- 🟢 Owner cannot set fees higher than 25%
- 🟢 Owner cannot change max transaction amount
- 🟡 Owner can exclude from fees
- 🟡 Owner can pause the contract
- 🔴 Owner can blacklist addresses

⚠️ Owner can withdraw funds sent to the contract address

⚠️ Owner can change the router address

Extra notes by the team

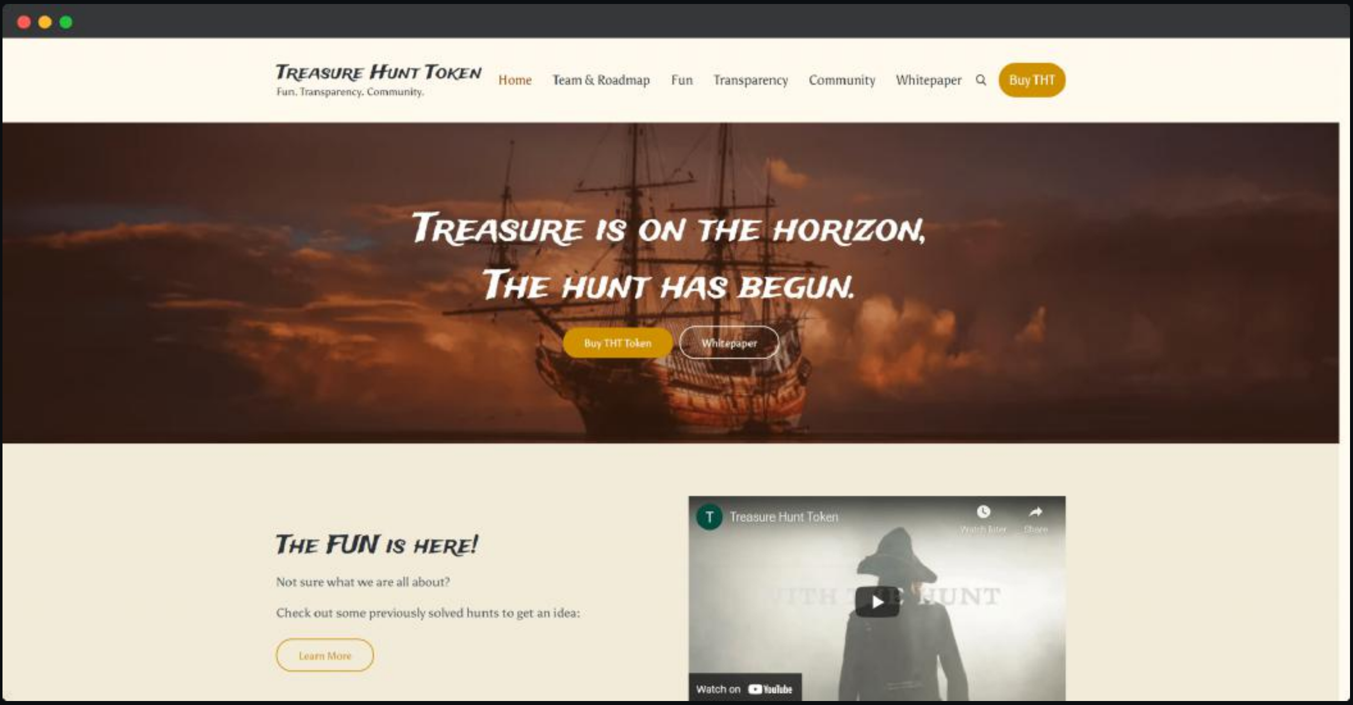
No notes

Contract Snapshot

```
contract TreasureHunt is Context, IERC20, IERC20Metadata, Ownable {
using SafeMath for uint256;

string private _name;
string private _symbol;

uint256 private _totalSupply;

mapping(address =&gt; uint256) private _balances;
mapping(address =&gt; mapping(address =&gt; uint256)) private _allowances;

////////////////////////////////////////////////////////////////////////////////////////////////////

struct feeRateStruct {
    uint256 treasureNFee;
    uint256 liquidityFee;
    uint256 treasureFee;
    uint256 marketingFee;
    uint256 operationFee;

}

feeRateStruct public buyFeeRates = feeRateStruct(
    {
        treasureNFee: 100,
        liquidityFee: 200,
        treasureFee: 200,
        marketingFee: 200,
        operationFee:200
    }
);

feeRateStruct public sellFeeRates = feeRateStruct(
    {
        treasureNFee: 100,
        liquidityFee: 400,
        treasureFee: 400,
```

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.



Treasure is on the horizon, The hunt has begun.

The FUN is here!

Not sure what we are all about?

Check out some previously solved hunts to get an idea:

● Mobile Friendly

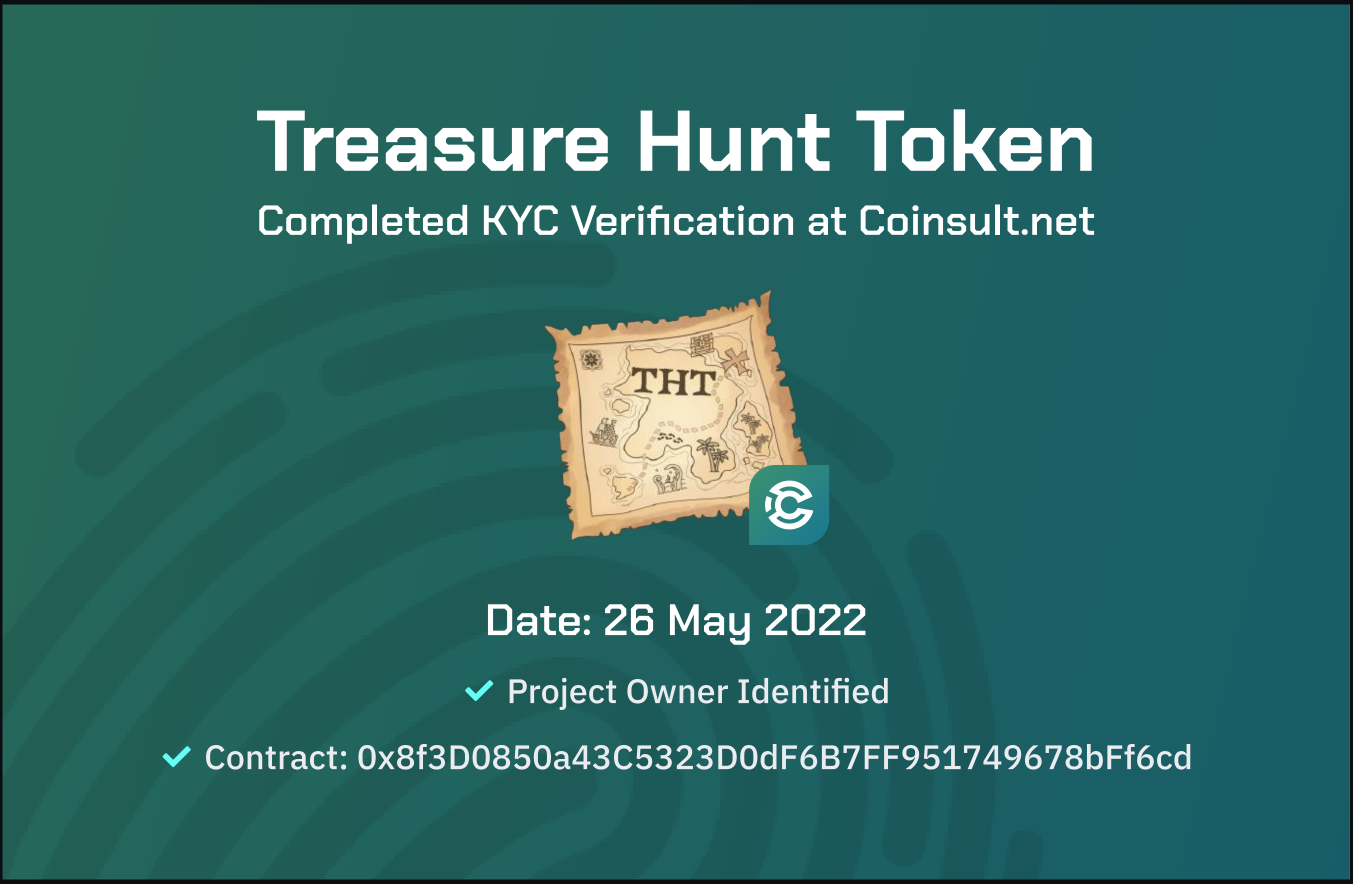
● Does not contain jQuery errors

● SSL Secured

● No major spelling errors

Project Overview

● KYC verified by Coinsult



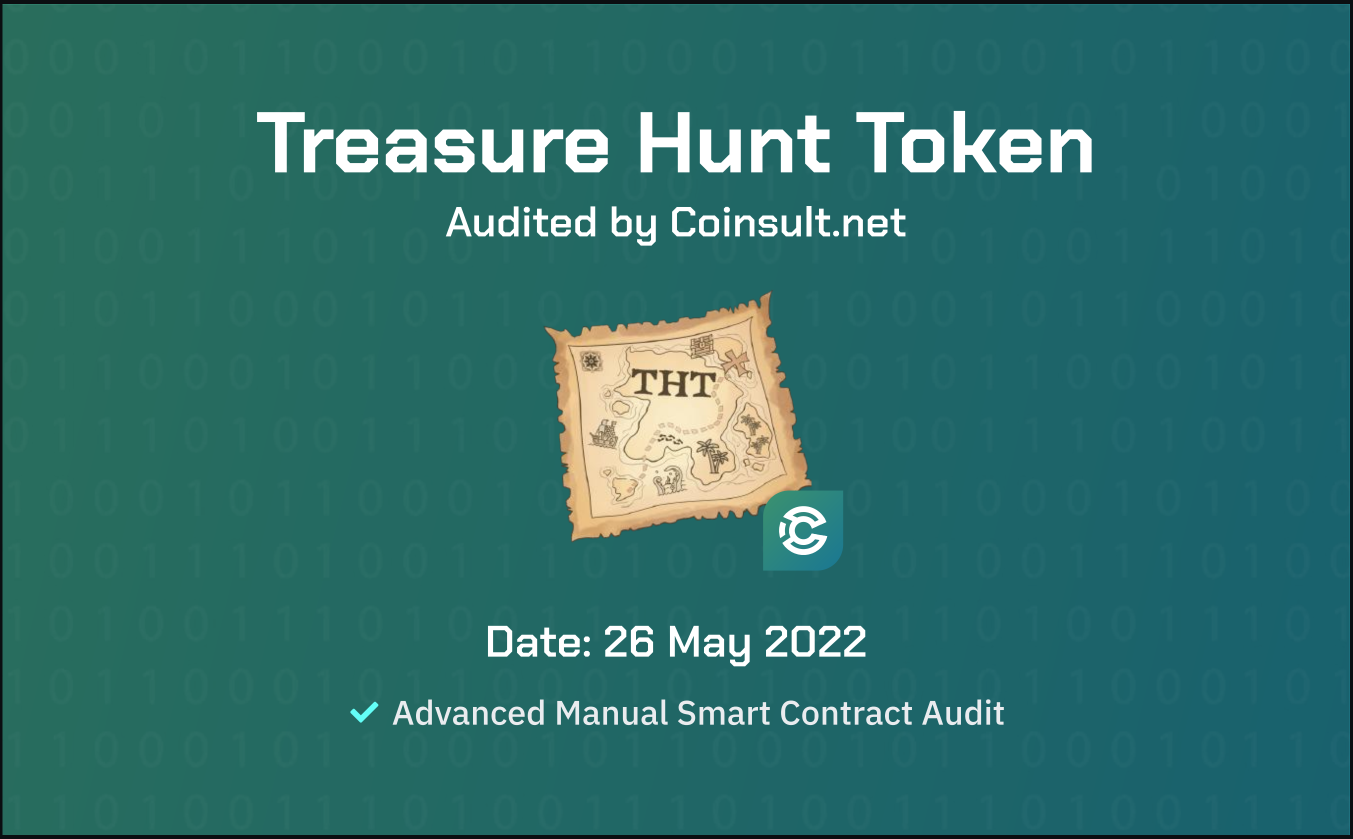
Treasure Hunt Token

Completed KYC Verification at Coinsult.net

Date: 26 May 2022

✓ Project Owner Identified

✓ Contract: 0x8f3D0850a43C5323D0dF6B7FF951749678bFf6cd



Treasure Hunt Token

Audited by Coinsult.net

Date: 26 May 2022

✓ Advanced Manual Smart Contract Audit