Coinsult

Advanced Manual
Smart Contract Audit

Project: Apollo Ventures
Website: https://a11.ventures

🟢 **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

0x1589833AAcF6D701821FEa04491A20BF44818F98

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

Not deployed yet

Source code

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

https://testnet.bscscan.com/address/0x703E84584F5aE5A366EEc3924FFc8eb054f1e307#code

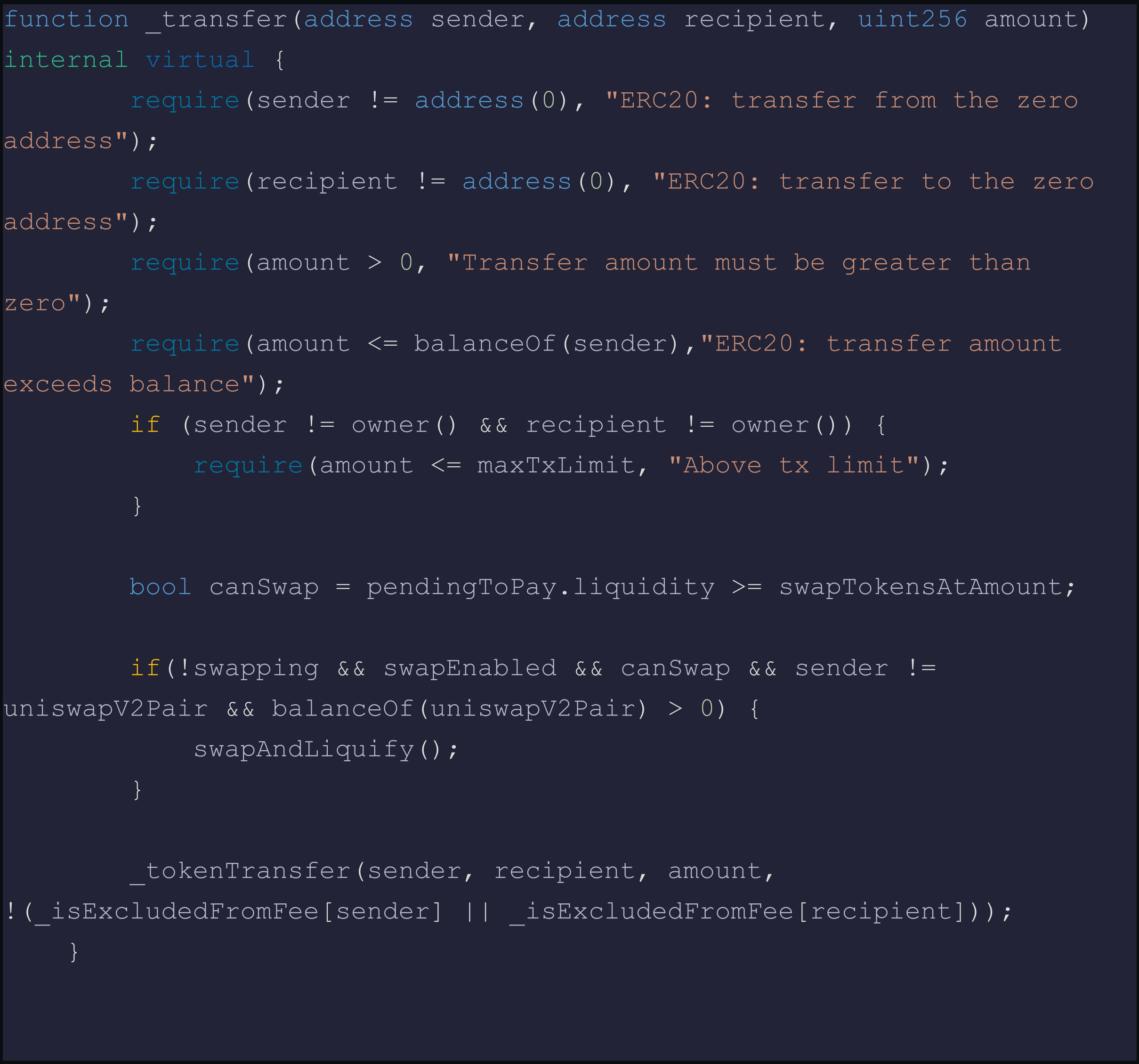
Note: Deployed on the testnet, now on main net: 0x1589833AAcF6D701821FEa04491A20BF44818F98

Manual Code Review

🟢 Low-risk

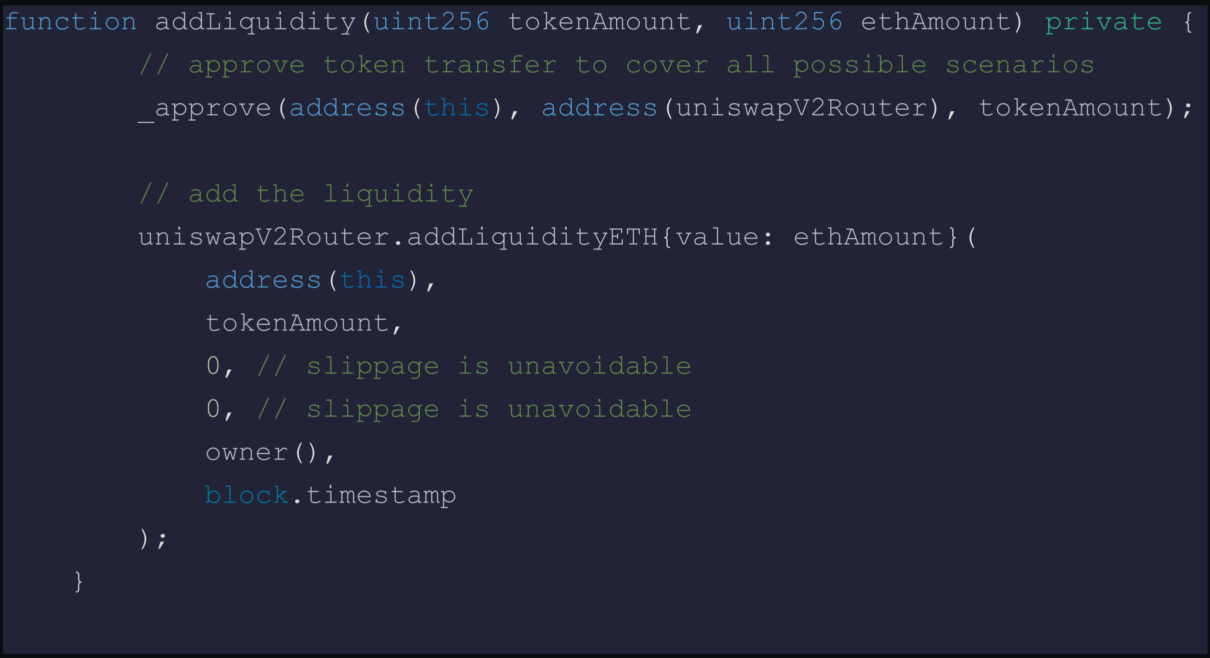
5 low-risk code issues found.
Could be fixed, will not bring problems.

- 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 virtual {
        require(sender != address(0), "ERC20: transfer from the zero
address");
        require(recipient != address(0), "ERC20: transfer to the zero
address");
        require(amount > 0, "Transfer amount must be greater than
zero");
        require(amount <= balanceOf(sender),"ERC20: transfer amount
exceeds balance");
        if (sender != owner() && recipient != owner()) {
            require(amount <= maxTxLimit, "Above tx limit");
        }

        bool canSwap = pendingToPay.liquidity >= swapTokensAtAmount;

        if(!swapping && swapEnabled && canSwap && sender !=
uniswapV2Pair && balanceOf(uniswapV2Pair) > 0) {
            swapAndLiquify();
        }

        _tokenTransfer(sender, recipient, amount,
!(_isExcludedFromFee[sender] || _isExcludedFromFee[recipient]));
    }
```

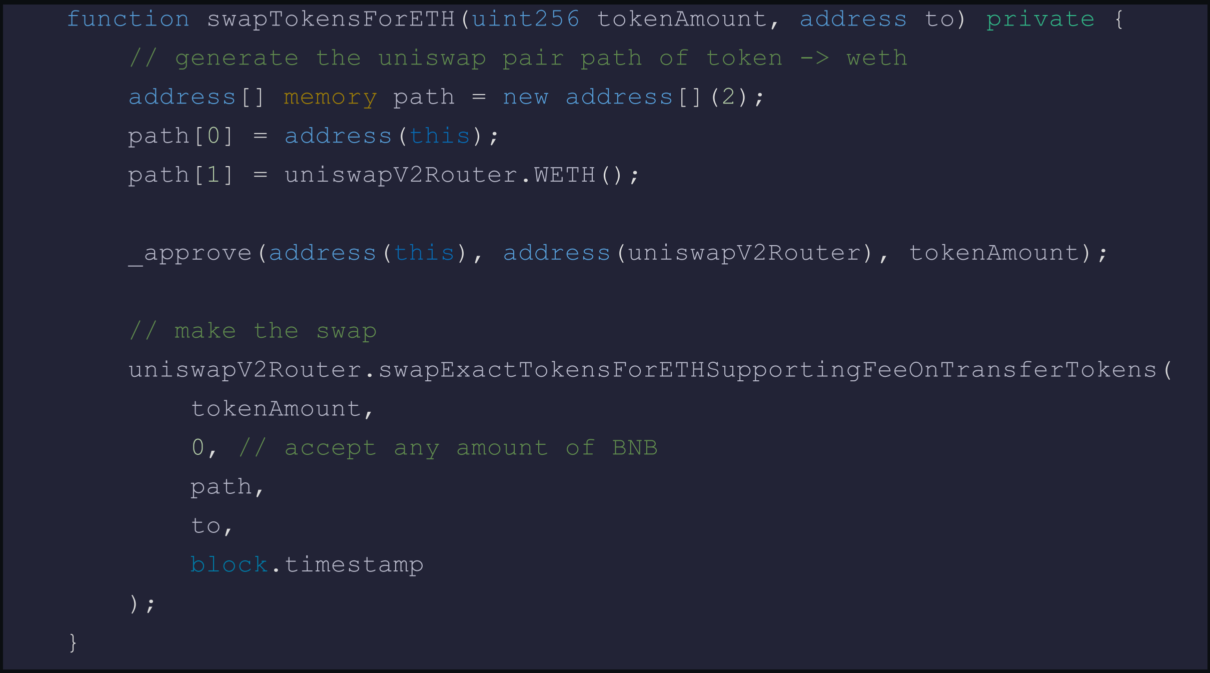
- Function which sends eth to arbitrary destination
  Ensure that an arbitrary user cannot withdraw unauthorized funds. More information: Slither



```
function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {
        // approve token transfer to cover all possible scenarios
        _approve(address(this), address(uniswapV2Router), tokenAmount);

        // add the liquidity
        uniswapV2Router.addLiquidityETH{value: ethAmount}(
            address(this),
            tokenAmount,
            0, // slippage is unavoidable
            0, // slippage is unavoidable
            owner(),
            block.timestamp
        );
    }
```

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

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



```
function swapTokensForETH(uint256 tokenAmount, address to) private {
    // generate the uniswap pair path of token -> weth
    address[] memory path = new address[](2);
    path[0] = address(this);
    path[1] = uniswapV2Router.WETH();

    _approve(address(this), address(uniswapV2Router), tokenAmount);

    // make the swap
    uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
        tokenAmount,
        0, // accept any amount of BNB
        path,
        to,
        block.timestamp
    );
}
```

- Missing zero address validation

  Check that the new address is not the zero address.

```
      marketingAddress = _marketingAddress;
      lotteryAddress = _lotteryAddress;
      NFTAppAddress= _NFTAppAddress;
```

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

  Use a local variable to hold the loop computation result.

```
function includeInReward(address account) external onlyOwner() {
        require(_isExcludedFromReward[account], "Account is not
excluded");
        for (uint256 i = 0; i < _excludedFromReward.length; i++) {
            if (_excludedFromReward[i] == account) {
                _excludedFromReward[i] =
_excludedFromReward[_excludedFromReward.length - 1];
                _tOwned[account] = 0;
                _isExcludedFromReward[account] = false;
                _excludedFromReward.pop();
                break;
            }
        }
    }
```

🟡 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 change fees but not more than 26%.

🟡 A lot of commented code is in the contract, this could be removed to increase the readability.

🟡 The ownership is not renounced.

🟡 Owner can whitelist addresses from fee.

🟡 No liquidity router hard coded in the contract.

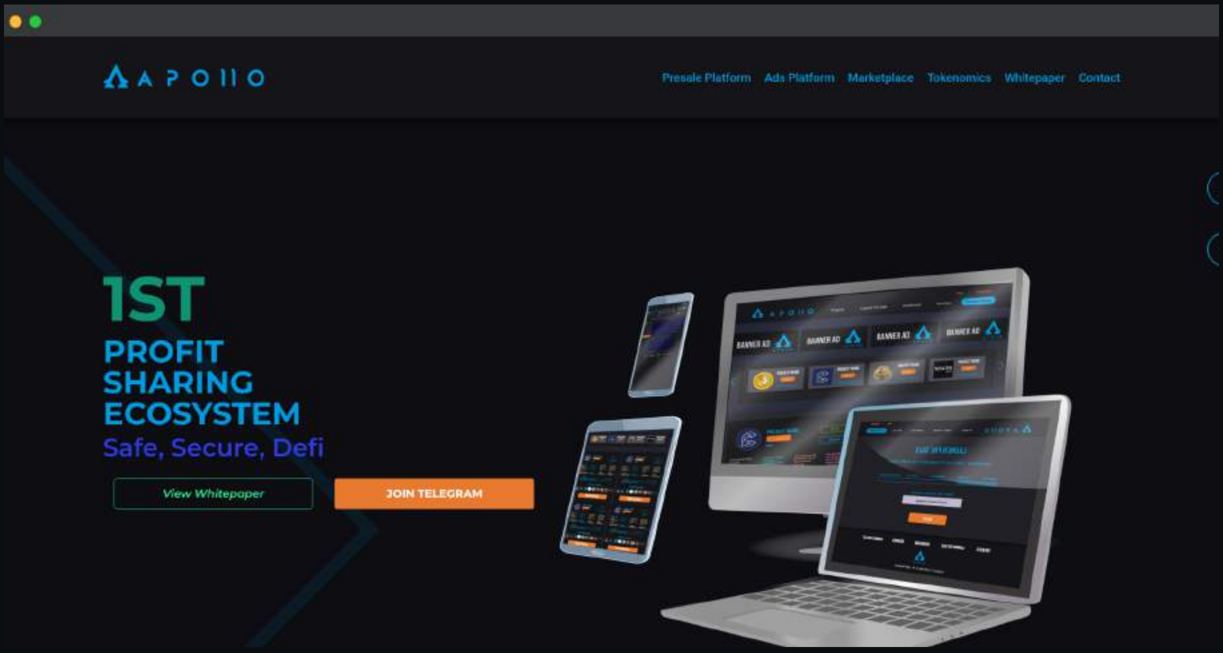
🟡 `SafeMath` is generally not needed starting with Solidity 0.8, since the compiler now has built in overflow checking.

🔴 Owner can pause and unpause the contract

Contract Snapshot

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

    address public marketingAddress;
    address public lotteryAddress;
    address public NFTAppAddress;
    address public uniswapV2Pair;
    address[] private _excludedFromReward;

    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 _isExcludedFromReward;

    uint256 private constant MAX = ~uint256(0);
    uint256 private constant _tTotal = 150_000_000 * 10**9;
    uint256 private _rTotal = (MAX - (MAX % _tTotal));
    uint256 public swapTokensAtAmount = 100_000 * 10**9;
    uint256 public maxTxLimit = 500_000 * 10**9;

    string private constant _name = "Apollo Ventures";
    string private constant _symbol = "A11";
    uint8 private constant _decimals = 9;

    bool private swapping;
    bool public swapEnabled;
    bool public tradingEnabled;
```

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
- 🟢 Contains no jQuery errors
- 🟢 SSL Secured
- 🟢 No major spelling errors

Loading speed: 86%

Rug-pull Review

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

🟢 Locked Liquidity - Not applicable

🟢 Large unlocked wallets - Not applicable

🟢 Doxxed Team

Honeypot Review

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

🟢 Ability to sell - Not applicable

🔴 Owner is able to pause the contract

🟢 Router not hard coded in the contract - Not applicable

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.