Coinsult

Advanced Manual Smart Contract Audit

Project: Aretis – Draft report

Website: https://aretis.io/

🟢 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

0x5326fCDf8ae618db847D1C6689A848b6D393763D testnet

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

Testnet

Source Code

Coinsult was comissioned by Aretis – Draft report to perform an audit based on the following smart contract:

https://testnet.bscscan.com/address/0x5326fCDf8ae618db847D1C6689A848b6D393763D

Testnet Contract

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 _transferFrom(
    address sender,
    address recipient,
    uint256 amount
) internal returns (bool) {
    require(!blacklist[sender] &amp;&amp; !blacklist[recipient], 'in-blacklist');

    if (inSwap) {
        return _basicTransfer(sender, recipient, amount);
    }
    if (shouldRebase()) {
        rebase();
    }

    if (shouldAddLiquidity()) {
        addLiquidity();
    }

    if (shouldSwapBack()) {
        swapBack();
    }
```

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.

Avoid relying on block.timestamp

block.timestamp can be manipulated by miners.

```
if (recipient == pair || sender == pair) {
    //sell token
    require(startTradingTime &gt; 0 &amp;&amp; block.timestamp &gt;= startTradingTime, 'can not trad
    if (block.timestamp &lt;= startTradingTime + 6) {
        _totalFee = _totalFee.add(_robotsFee);
        _gonBalances[autoLiquidityReceiver] = _gonBalances[autoLiquidityReceiver].add(
            gonAmount.div(feeDenominator).mul(_robotsFee)
        );
    }
}
```

Recommendation

Do not use `block.timestamp`, now or `blockhash` as a source of randomness

Exploit scenario

```
contract Game {

    uint reward_determining_number;

    function guessing() external{
      reward_determining_number = uint256(block.blockhash(10000)) % 10;
    }
}
```

Eve is a miner. Eve calls `guessing` and re-orders the block containing the transaction. As a result, Eve wins the game.

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

Missing events arithmetic

Detect missing events for critical arithmetic parameters.

```
function setStartTradingTime(uint256 _time) public onlyOwner {
    startTradingTime = _time;
    if (_time &gt; 0) {
        _lastAddLiquidityTime = _time;
        if (_lastRebasedTime == 0) {
            _lastRebasedTime = _time;
        }
    }
}
```

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.

```
if (recipient == pair &amp;&amp; _isFeeExempt[sender] == false &amp;&amp; _isFeeExempt[recipient] ==
    //only can sell 99% of balance
    if (gonAmount &gt;= _gonBalances[sender].div(1000).mul(999)) {
        gonAmount = _gonBalances[sender].div(1000).mul(999);
    }
    //require(gonAmount&lt;=_gonBalances[sender].mul(99).div(100),&quot;only can sell 99% of balance&
}
```

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`.

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

Conformance to Solidity naming conventions

Allow _ at the beginning of the mixed_case match for private variables and unused parameters.

```
uint256 private TOTAL_GONS;

uint256 private constant MAX_SUPPLY = ~uint128(0) / 1e14;
```

Recommendation

Follow the Solidity naming convention.

Rule exceptions

- Allow constant variable name/symbol/decimals to be lowercase (`ERC20`).
- Allow _ at the beginning of the `mixed_case` match for private variables and unused parameters.

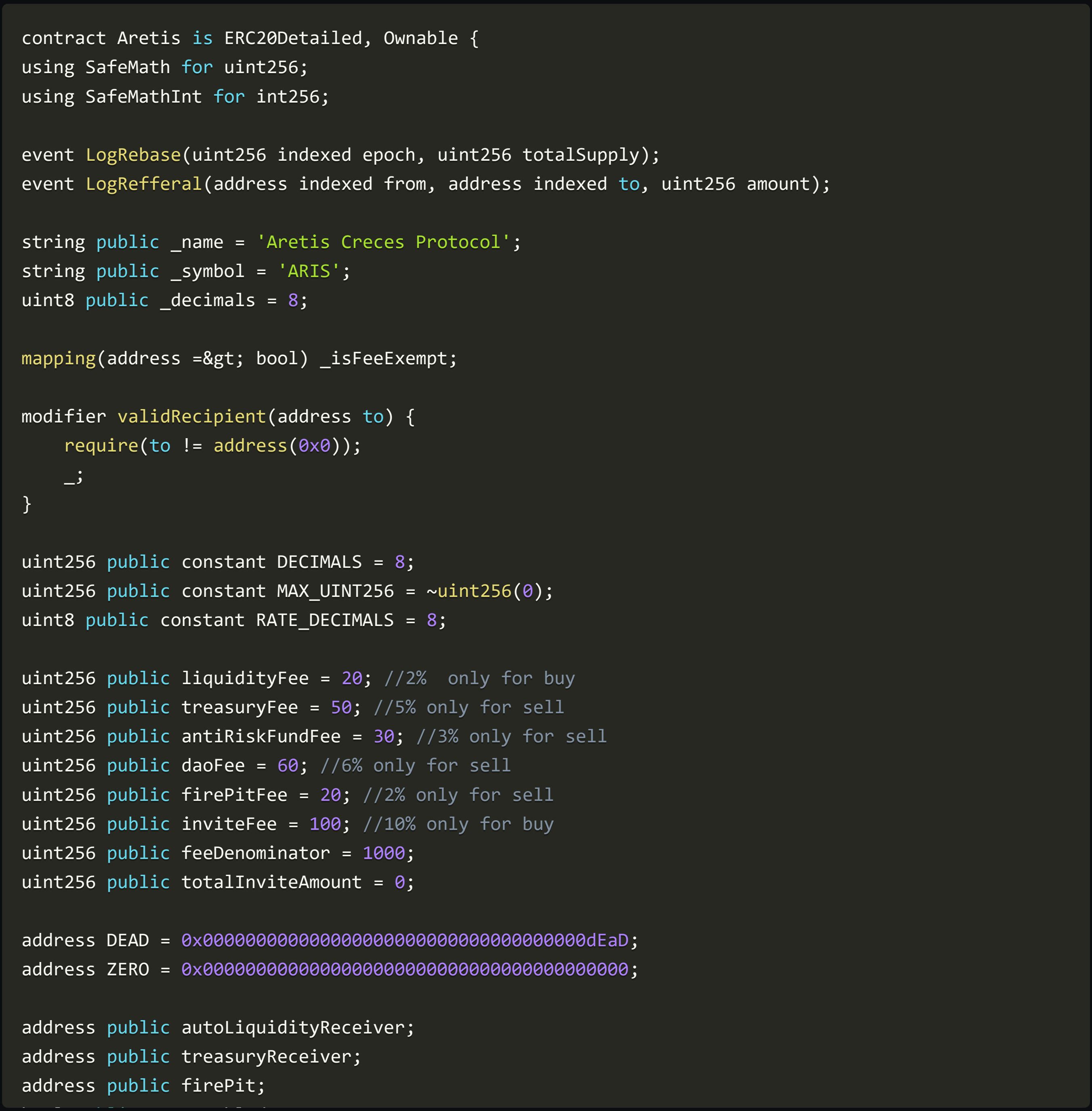
Owner privileges

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

Extra notes by the team

No notes

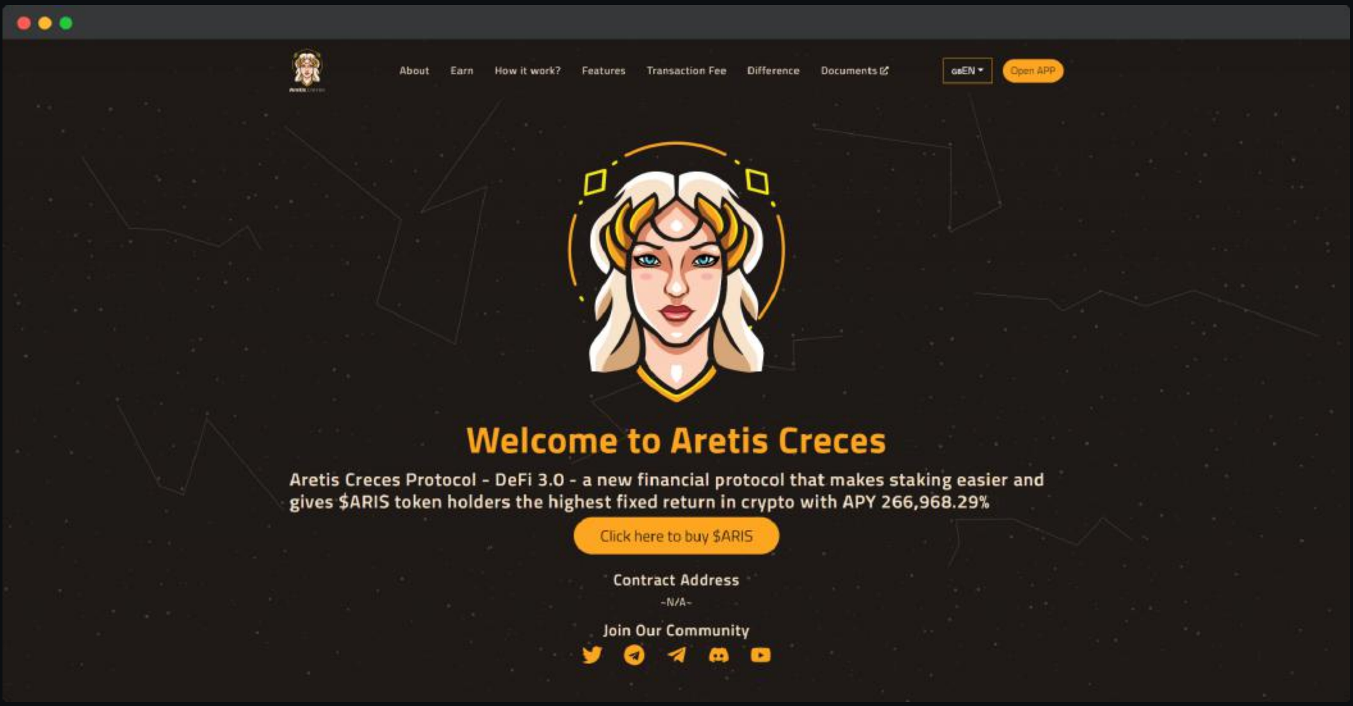
Contract Snapshot



```
contract Aretis is ERC20Detailed, Ownable {
using SafeMath for uint256;
using SafeMathInt for int256;

event LogRebase(uint256 indexed epoch, uint256 totalSupply);
event LogRefferal(address indexed from, address indexed to, uint256 amount);

string public _name = 'Aretis Creces Protocol';
string public _symbol = 'ARIS';
uint8 public _decimals = 8;

mapping(address =&gt; bool) _isFeeExempt;

modifier validRecipient(address to) {
    require(to != address(0x0));
    _;
}

uint256 public constant DECIMALS = 8;
uint256 public constant MAX_UINT256 = ~uint256(0);
uint8 public constant RATE_DECIMALS = 8;

uint256 public liquidityFee = 20; //2%  only for buy
uint256 public treasuryFee = 50; //5% only for sell
uint256 public antiRiskFundFee = 30; //3% only for sell
uint256 public daoFee = 60; //6% only for sell
uint256 public firePitFee = 20; //2% only for sell
uint256 public inviteFee = 100; //10% only for buy
uint256 public feeDenominator = 1000;
uint256 public totalInviteAmount = 0;

address DEAD = 0x000000000000000000000000000000000000dEaD;
address ZERO = 0x0000000000000000000000000000000000000000;

address public autoLiquidityReceiver;
address public treasuryReceiver;
address public firePit;
```

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.



Welcome to Aretis Creces

Aretis Creces Protocol - DeFi 3.0 - a new financial protocol that makes staking easier and gives $ARIS token holders the highest fixed return in crypto with APY 266,968.29%

Contract Address

-N/A-

Join Our Community

- ● Mobile Friendly
- ● Does not contain jQuery errors
- ● SSL Secured
- ● No major spelling errors

Project Overview

● Not KYC verified by Coinsult

Aretis – Draft report

Audited by Coinsult.net

Date: 26 June 2022

✓ Advanced Manual Smart Contract Audit