HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: ArthCoin
Date: April 19th, 2021

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

This document may contain confidential information about IT systems and the intellectual property of the Customer as well as information about potential vulnerabilities and methods of their exploitation.

The report containing confidential information can be used internally by the Customer, or it can be disclosed publicly after all vulnerabilities are fixed - upon a decision of the Customer.

Document

Name	Smart Contract Code Review and Security Analysis Report for ArthCoin - Initial Audit
Approved by	Andrew Matiukhin | CTO Hacken OU
Type	StableCoin and Staking pool
Platform	Ethereum / Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
List of contracts	https://github.com/MahaDAO/arthcoin-v2/blob/master/contracts/Arth/Arth.sol https://github.com/MahaDAO/arthcoin-v2/blob/develop/contracts/Staking/StakingRewards.sol https://github.com/MahaDAO/arthcoin-v2/blob/master/contracts/Arth/Pools/ArthPool.sol https://github.com/MahaDAO/arthcoin-v2/blob/develop/contracts/Genesis/Genesis.sol https://github.com/MahaDAO/arthcoin-v2/blob/develop/contracts/Arth/ArthController.sol
Timeline	14 APRIL 2021 – 19 APRIL 2021
Changelog	19 APRIL 2021 – INITIAL AUDIT

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Table of contents

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Introduction

Hacken OÜ (Consultant) was contracted by ArthCoin (Customer) to conduct a Smart Contract Code Review and Security Analysis. This report presents the findings of the security assessment of Customer's smart contract and its code review conducted on April 19th, 2021.

Scope

The scope of the project is the following solidity contracts:

https://github.com/MahaDAO/arthcoin-v2/blob/master/contracts/Arth/Arth.sol
https://github.com/MahaDAO/arthcoin-v2/blob/develop/contracts/Staking/StakingRewards.sol
https://github.com/MahaDAO/arthcoin-v2/blob/master/contracts/Arth/Pools/ArthPool.sol
https://github.com/MahaDAO/arthcoin-v2/blob/develop/contracts/Genesis/Genesis.sol
https://github.com/MahaDAO/arthcoin-v2/blob/develop/contracts/Arth/ArthController.sol

We have scanned these smart contracts for commonly known and more specific vulnerabilities. Here are some of the commonly known vulnerabilities that are considered:

Category	Check Item
Code review	▪ Reentrancy ▪ Ownership Takeover ▪ Timestamp Dependence ▪ Gas Limit and Loops ▪ DoS with (Unexpected) Throw ▪ DoS with Block Gas Limit ▪ Transaction-Ordering Dependence ▪ Style guide violation ▪ Costly Loop ▪ ERC20 API violation ▪ Unchecked external call ▪ Unchecked math ▪ Unsafe type inference ▪ Implicit visibility level ▪ Deployment Consistency ▪ Repository Consistency ▪ Data Consistency

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Functional review	▪ Business Logics Review ▪ Functionality Checks ▪ Access Control & Authorization ▪ Escrow manipulation ▪ Token Supply manipulation ▪ Asset’s integrity ▪ User Balances manipulation ▪ Kill-Switch Mechanism ▪ Operation Trails & Event Generation

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Executive Summary

According to the assessment, the Customer's smart contract is secured.

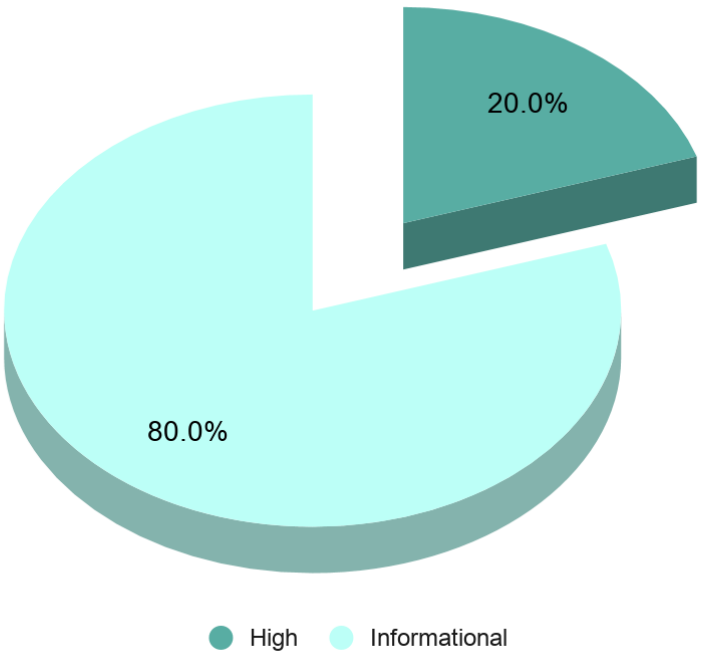
Insecure	Poor secured	Secured	Well-secured

You are here

Our team performed an analysis of code functionality, manual audit, and automated checks with Mythril and Slither. All issues found during automated analysis were manually reviewed, and important vulnerabilities are presented in the Audit overview section. A general overview is presented in AS-IS section, and all found issues can be found in the Audit overview section.

Security engineers found **1** high and **4** informational issues during the first review.

Graph 1. The distribution of vulnerabilities after the first review.



Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to assets loss or data manipulations.
High	High-level vulnerabilities are difficult to exploit; however, they also have a significant impact on smart contract execution, e.g., public access to crucial functions
Medium	Medium-level vulnerabilities are important to fix; however, they can't lead to assets loss or data manipulations.
Low	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that can't have a significant impact on execution
Lowest / Code Style / Best Practice	Lowest-level vulnerabilities, code style violations, and info statements can't affect smart contract execution and can be ignored.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Audit overview

Critical

No Critical severity issues were found.

High

1. **Vulnerability:** Uninitialized State Variable
   Contracts: StakingRewards

   Accessing state variable which never goes initialized

   Recommendation: Please consider initializing **_arthController** with a value on the constructing stage

   Lines: StakingRewards.sol#453-465

```
function crBoostMultiplier() public view returns (uint256) {
    uint256 multiplier =
        uint256(_MULTIPLIER_BASE).add(
            (
                uint256(_MULTIPLIER_BASE).sub(
                    _arthController.getGlobalCollateralRatio()
                )
            )
                .mul(crBoostMaxMultiplier.sub(_MULTIPLIER_BASE))
                .div(_MULTIPLIER_BASE)
        );
    return multiplier;
}
```

Medium

No Medium severity issues were found.

Low

No Low severity issues were found.

Lowest / Code style / Best Practice

1. **Vulnerability:** Too many digits
   Contracts: ARTHStablecoin, ArthController, StakingRewards

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Literals with many digits are difficult to read and review. Please consider using *scientific notation* and *ether units* for better readability. Ex. instead of 2000000e18 try the following:

- 2e6 ether
- 2_000_000e18
- 2_000_000 ether

Note: Seen in the FRAX. Not Fixed.

Lines: Arth.sol#23

```
uint256 public constant  genesisSupply = 22000000e18; // 22M ARTH
(testnet) & 5k (Mainnet).
```

Lines: ArthController.sol#70

```
uint256 public constant genesisSupply = 2_000_000e18; // 2M ARTH
(testnet) & 5k (Mainnet).
```

Lines: StakingRewards.sol#69

```
uint256 public lockedStakeMaxMultiplier = 3000000; // 6 decimals of
precision. 1x = 1000000
```

2. **Vulnerability:** State variables that could be declared immutable
Contract: Genesis, StakingRewards

State variables that never change their values which is assigned in the constructor should be declared immuatble to save gas.

Note: Some were seen in the FRAX. Not Fixed.

Lines: Genesis.sol#24-26

```
IWETH private _WETH;
IARTH private _ARTH;
IARTHX private _ARTHX;
```

Lines: Genesis.sol#28-29

```
IERC20Mintable private _MAHA;
IUniswapV2Router02 private _ROUTER;
```

Lines: StakingRewards.sol#52-58

```
IERC20 public rewardsToken;
IERC20 public stakingToken;
```

```
IARTH private _ARTH;
IARTHController private _arthController;

// This staking pool's percentage of the total ARTHX being distributed
by all pools, 6 decimals of precision
uint256 public poolWeight;
```

3. **Vulnerability:** Unused State Variable
   Contracts: StakingRewards, ArthController

   Private state variable is assigned but never used in the contract, and also uninitialized public state variable

   Lines: StakingRewards.sol#54, 160

```
IARTH private _ARTH;

/ ... skipped ... /

_ARTH = IARTH(_arthAddress);
```

   Lines: ArthController.sol#32

```
IERC20 public ARTHX;
```

4. **Vulnerability:** Public function that could be declared external
   Contracts: ArthController, StakingRewards, Genesis

   public functions that are never called by the contract should be declared **external** to save gas.

   Note: Some were seen in the FRAX. Not Fixed.

   Lines: ArthController.sol#226-230

```
function setGlobalCollateralRatio(uint256 _globalCollateralRatio)
    public
    override
    onlyAdmin
{
```

   Lines: ArthController.sol#234-238

```
function setARTHXAddress(address _arthxAddress)
    public
    override
    onlyByOwnerOrGovernance
{
```

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Lines: ArthController.sol#242-246

```
function setPriceTarget(uint256 newPriceTarget)
   public
   override
   onlyByOwnerOrGovernance
{
```

Lines: ArthController.sol#258-262

```
function setETHGMUOracle(address _ethGMUConsumerAddress)
   public
   override
   onlyByOwnerOrGovernance
{
```

Lines: ArthController.sol#268-271

```
function setARTHXETHOracle(
   address _arthxOracleAddress,
   address _wethAddress
) public override onlyByOwnerOrGovernance {
```

Lines: ArthController.sol#277-281

```
function setARTHETHOracle(address _arthOracleAddress, address
_wethAddress)
   public
   override
   onlyByOwnerOrGovernance
{
```

Lines: ArthController.sol#287

```
function toggleCollateralRatio() public override
onlyCollateralRatioPauser {
```

Lines: ArthController.sol#291-295

```
function setMintingFee(uint256 fee)
   public
   override
   onlyByOwnerOrGovernance
{
```

Lines: ArthController.sol#299-303

```
function setArthStep(uint256 newStep)
   public
   override
   onlyByOwnerOrGovernance
```

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

```
{
```

Lines: ArthController.sol#307-311

```
function setRedemptionFee(uint256 fee)
    public
    override
    onlyByOwnerOrGovernance
{
```

Lines: ArthController.sol#315-319

```
function setOwner(address _ownerAddress)
    public
    override
    onlyByOwnerOrGovernance
{
```

Lines: ArthController.sol#323-327

```
function setPriceBand(uint256 _priceBand)
    public
    override
    onlyByOwnerOrGovernance
{
```

Lines: ArthController.sol#331-335

```
function setTimelock(address newTimelock)
    public
    override
    onlyByOwnerOrGovernance
{
```

Lines: ArthPool.sol#801

```
function getCollateralGMUBalance() public view override returns
(uint256) {
```

Lines: StakingRewards.sol#468

```
function lockedBalanceOf(address account) public view returns (uint256)
```

Lines: StakingRewards.sol#493

```
function getReward() public override nonReentrant
updateReward(msg.sender) {
```

Lines: Genesis.sol#194

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

```
function redeem(uint256 amount) public {
```

Lines: Genesis.sol#203

```
function distribute() public onlyOwner hasEnded {
```

Lines: Genesis.sol#219

```
function getIsRaisedBetweenCaps() public view returns (bool) {
```

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Conclusion

Smart contracts within the scope were manually reviewed and analyzed with static analysis tools. For the contract, high-level description of functionality was presented in As-Is overview section of the report.

Audit report contains all found security vulnerabilities and other issues in the reviewed code.

Security engineers found **1** high and **4** informational issues during the first review.

Category	Check Items	Comments
Functional Review	Business Logics Review	➔ Uninitialized State Variable
Code Review	Style guide violation	➔ Public function that could be declared external ➔ State variables that could be declared immutable ➔ Too many digits ➔ Unused State Variable

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed in accordance with the best industry practices at the date of this report, in relation to cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report (Source Code); the Source Code compilation, deployment, and functionality (performing the intended functions).

The audit makes no statements or warranties on security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bugfree status or any other statements of the contract. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only - we recommend proceeding with several independent audits and a public bug bounty program to ensure security of smart contracts.

Technical Disclaimer

Smart contracts are deployed and executed on blockchain platform. The platform, its programming language, and other software related to the smart contract can have its vulnerabilities that can lead to hacks. Thus, the audit can't guarantee the explicit security of the audited smart contracts.