HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: DAO Land
Date: December 7th, 2021

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

Document

Name	Smart Contract Code Review and Security Analysis Report for DAO Land.
Approved by	Andrew Matiukhin | CTO Hacken OU
Type	Staking contracts
Platform	Ethereum / Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Repository	1. https://github.com/DaoLand/DAOLand-liquidity-farm 2. https://github.com/DaoLand/DAOLand-mining
Commit	1. DAOLAND-LIQUIDITY-FARM: 8F7418716330E2FA2746CFCF198851A48847EDE5 2. DAOLAND-MINING: 5B185D9153C6643ECCB39BC2EC3B7C8B59C38548
Technical Documentation	NO
JS tests	NO
Website	daoland.io
Timeline	30 NOVEMBER 2021 - 3 DECEMBER 2021
Changelog	3 DECEMBER 2021 - INITIAL AUDIT 7 DECEMBER 2021 - SECOND REVIEW

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

Table of contents

Introduction 4

Scope 4

Executive Summary 5

Severity Definitions 8

Audit overview 9

Conclusion 12

Disclaimers 13

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

Introduction

Hacken OÜ (Consultant) was contracted by DAO Land (Customer) to conduct a Smart Contract Code Review and Security Analysis. This report presents the findings of the security assessment of the Customer's smart contract and its code review conducted between November 30th, 2021 - December 3rd, 2021.

Second review conducted on December 7th, 2021.

Scope

The scope of the project is smart contracts in the repository:

Repository:
https://github.com/DaoLand/DAOLand-liquidity-farm
https://github.com/DaoLand/DAOLand-mining
Commit:
DAOLand-liquidity-farm: 8f7418716330e2fa2746cfcf198851a48847ede5
DAOLand-mining: 5b185d9153c6643eccb39bc2ec3b7c8b59c38548
Post-audit:
DAOLand-liquidity-farm: e0d4841f4447d467f42c9b7ad0d60e66b25245be
DAOLand-mining: f87fe7d795f99ceea7ca210b4db9478a89013c82
Technical Documentation: No
JS tests: No
Contracts:
Staking.sol
Farming.sol

We have scanned this smart contract 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
Functional review	▪ Business Logics Review ▪ Functionality Checks ▪ Access Control & Authorization ▪ Escrow manipulation ▪ Token Supply manipulation ▪ Assets integrity ▪ User Balances manipulation ▪ Data Consistency manipulation ▪ Kill-Switch Mechanism ▪ Operation Trails & Event Generation

Executive Summary

According to the assessment, the Customer's smart contracts are well-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. All found issues can be found in the Audit overview section.

As a result of the audit, security engineers found **1** high, **2** medium and **4** low severity issues.

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

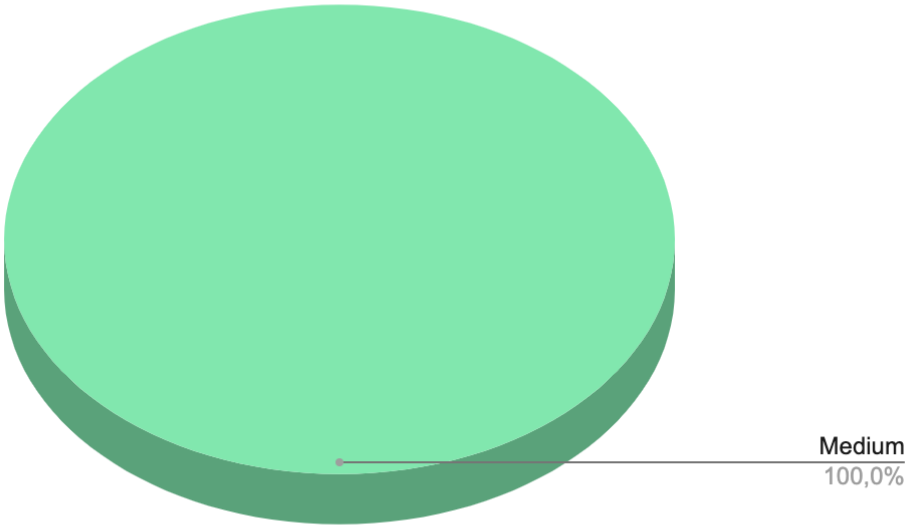
As a result of the second review, security engineers found **1** medium severity issue.

Notice:

The admin for Satking and Farming contracts has an ability to withdraw any token at any time including currently staked tokens.

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

Graph 1. The distribution of vulnerabilities after the audit.



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

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

Audit overview

Critical

No critical issues were found.

High

Funds stay locked after the staking period is over.

After the staking period is over and after the enough time passed, the condition “2**halvingPeriodsQuantity <= startingRewardsPerEpoch” (line, 372, _produce()) becomes false and the transaction will revert. Thus, function update() will revert, and function unstake() will revert. User’s funds will become locked.

Contracts:Staking.sol, Farming.sol

Function: unstake(), _produce()

Recommendation: review the logic and proceed with the return in the _produce() function in case if the staking is over.

Status: Fixed.

Medium

1. Overpowered admin.

Admin has an ability to withdraw any token from the contract, including deposited and reward token. This can be helpful for the unexpected locks or issues, nevertheless it allows the withdrawal of all user’s tokens.

Contracts:Staking.sol, Farming.sol

Function: withdrawToken()

Recommendation: add conditions to protect user’s funds and allow the withdrawal of the deposited token only back to the depositors address. Also it is preferable to have limits for the deposited token withdrawal only after the end of the staking.

Status: Not fixed.

2. Double call of the update.

update() function is called twice in the stake() function - once before the storage update and once after it. Though update() function depends on the _produce() function which depends only on the block timestamp. Thus - on the second call nothing happens, because timestamp does not change. So it is a lot of gas spent in the single function, which makes function expensive for the user.

Contracts:Staking.sol, Farming.sol

Function: stake()

Recommendation: Remove extra update() call.

Status: Fixed.

Low

1. No need in the additional role.

The role does not perform any additional actions or actions different from which the default admin role is capable of. Thus it makes the role usage and changing the admin role unnecessary.

Contracts:Staking.sol, Farming.sol

Function: ADMIN_ROLE constant

Recommendation: consider using the default admin role for the gas consumption decreasement during the deployment.

Status: Fixed.

2. Add events for admin functions.

Add informative events for all crucial parameters change

Contracts:Staking.sol, Farming.sol

Recommendation: consider addition of events for core parameters change.

Status: Fixed.

3. Move the number to the constant.

Move “magic number” 1e20 to the public documented constant

Contracts: Staking.sol

Function: stake(), line 201; unstake(), line 221, update(), line 301
The same applies to the analogue functions in the Farming.sol contract

Recommendation: create public constant or use “precision” constant

Status: Fixed.

4. Check for the fine percent.

Contracts: Staking.sol, Farming.sol

Function: changeFinePercentParam()

Recommendation: add check for the percent to be not higher than the precision.

Status: Fixed.

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.

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

As a result of the audit, security engineers found **1** high, **2** medium and **4** low severity issues.

As a result of the second review, security engineers found **1** medium severity issue.

Notice:

The admin for Satking and Farming contracts has an ability to withdraw any token at any time including currently staked tokens.

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 the security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bug-free 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 the security of smart contracts.

Technical Disclaimer

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