HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: Minto
Date: July 12th, 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 Minto - Second Review
Approved by	Andrew Matiukhin | CTO Hacken OU
Type	ERC20 Burnable Lockable, Staking
Platform	Ethereum / Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Deployed contracts	https://hecoinfo.com/address/0x410a56541bD912F9B60943fcB344f1E3D6F09567 https://hecoinfo.com/address/0x9Cad4215FD0fc460B042eC86AbDe0130aA77069E
Timeline	28 JUNE 2021 - 12 JULY 2021
Changelog	30 JUNE 2021 - INITIAL AUDIT 12 JULY 2021 - SECOND REVIEW

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 Minto (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 July 12th, 2021.

Scope

The scope of the project is the next smart contracts:

Deployed contracts:
BTCMT - https://hecoinfo.com/address/0x410a56541bD912F9B60943fcB344f1E3D6F09567#readContract
Staking - https://hecoinfo.com/address/0x9Cad4215FD0fc460B042eC86AbDe0130aA77069E#readContract

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 contracts are Well-secured.

Insecure	Poor secured	Secured	Well-secured

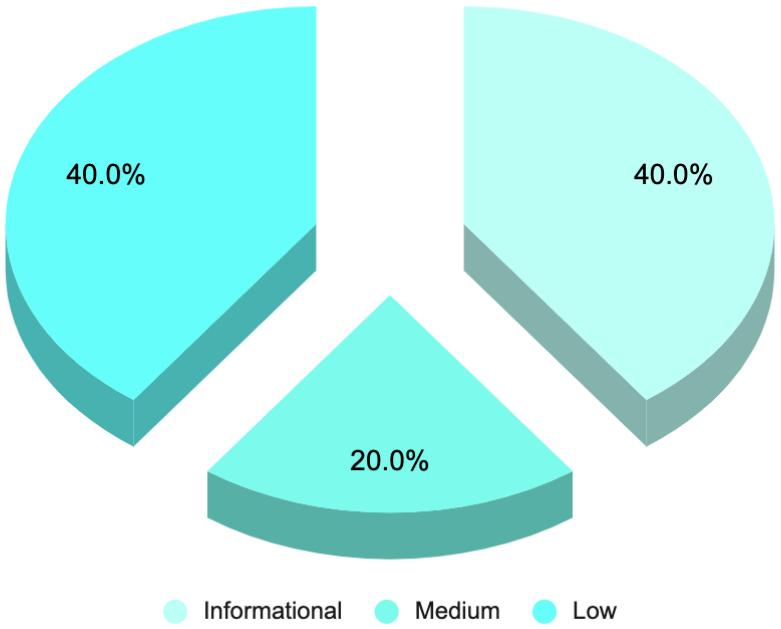
You are here

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.

Security engineers found 1 medium, 2 low and 2 informational issues during the first review.

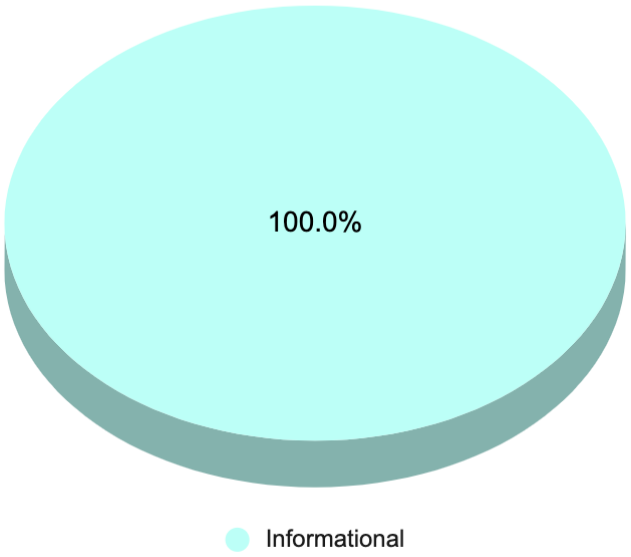
Security engineers found 1 informational issue during the second review.

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

Graph 2. The distribution of vulnerabilities after the second 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

No High severity issues were found.

Medium

1. Vulnerability: Some of tests provided fail

   Some of tests which was written by developers and provided to audit failing

   Recommendation: Please check tests and fix the

   Fixed before second review. Now all tests are passing

Low

1. Vulnerability: Centralization / Privilege

   Across contracts there are some roles with higher privileges like: DEFAULT_ADMIN_ROLE, MINTER_ROLE for BTCMT token and Owner for Staking. Each of those roles could modify critical configurations. If an attacker could ever get control of any of those addresses, they could perform actions which could cause users' funds loss

   Also, having such powerful addresses causes the centralization which could have actions done without community decision.

   Recommendation: renounce the ownership or transfer it to Timelock with multisig governance contract. This will let users feel safe and monitor any changes.

2. Vulnerability: No event on farm add/remove

   It is the best practice to emit events on admin actions like adding and removing farms. That will allow users to follow those events and see when some are added or removed.

   Recommendation: Please consider emitting events on adding / removing farms

   Fixed before second review

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

Lowest / Code style / Best Practice

1. Vulnerability: Boolean equality

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

   Recommendation: Remove the equality to the boolean constant.

   Fixed before second review

2. Vulnerability: Public function that could be declared external

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

   Recommendation: Please consider using the external attribute for functions never called from the contract

   Fixed before second review

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.

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

Security engineers found 1 medium, 2 low and 2 informational issues during the first review.

Security engineers found 1 informational issue during the second review.

Category	Check Items	Comments
➔ Functional Review	➔ Centralization	➔ Centralization / Privilege

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 the 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.