HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: My Liquidity Partner
Date: April 26th, 2022

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 My Liquidity Partner.
Approved By	Evgeniy Bezuglyi | SC Department Head at Hacken OU
Type of Contracts	Swap
Platform	EVM
Language	Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Website	https://myliquiditypartner.com
Timeline	28.03.2022 - 21.04.2022
Changelog	30.03.2022 - Initial Review 19.04.2022 - 21.04.2022 - Remediation 26.04 - Second remediation

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 Ruby Technologies Ltd (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 contracts.

Scope

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

Initial review
Repository:
https://rinkeby.etherscan.io/address/0x269a37a38d7E86AeB02fD29e9e7011F7BACaECA6
https://rinkeby.etherscan.io/address/0xe265F652C20dEA3aaE82Ea5f1511f6B1C89bBc7f
Documentation: Yes
JS tests: No
Contracts:
PrivateRound.sol
MyLiquidityPool.sol
Second review
Repository:
https://github.com/DAO-VC/MyLiquidityPool
Commit:
283af42b62dbd5e5620251063b26b3f7e28eab13
Documentation: Yes
JS tests: Yes
Contracts:
contracts/rounds/PrivateRound.sol
contracts/token.sol
Third review
Repository:
https://github.com/DAO-VC/MyLiquidityPool
Commit:
489d591df02c0307ce4b9312460fd2dad44637fe
Documentation: Yes
JS tests: Yes
Contracts:
contracts/rounds/PrivateRound.sol
contracts/token.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 ▪ Transaction-Ordering Dependence ▪ Style guide violation ▪ EIP standards violation ▪ Unchecked external call ▪ Unchecked math ▪ Unsafe type inference ▪ Implicit visibility level ▪ Deployment Consistency ▪ Repository Consistency
Functional review	▪ Business Logics Review ▪ Functionality Checks ▪ Access Control & Authorization ▪ Escrow manipulation ▪ Token Supply manipulation ▪ Assets integrity ▪ User Balances manipulation ▪ Data Consistency ▪ Kill-Switch Mechanism

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

Executive Summary

The score measurements details can be found in the corresponding section of the methodology.

Documentation quality

The Customer provided functional requirements and tokenomics, superficial technical requirements and the detailed comments in the contracts. The total Documentation Quality score is **10** out of **10**.

Code quality

The total CodeQuality score is **8** out of **10**. Unit tests were provided, the test coverage for the PublicRound contract is low.

Architecture quality

The architecture quality score is **10** out of **10**. The architecture is clear.

Security score

As a result of the audit, security engineers found **4** high, and **1** medium, and **3** low severity issues. The security score is **0** out of **10**. All found issues are displayed in the “Issues overview” section.

As a result of the second review, security engineers found **1** new critical, **1** high, and **2** low severity issues. The code contains **1** critical, **1** high, and **2** low severity issues. The security score is **0** out of **10**.

As a result of the third review, security engineers found **no** new critical severity issues. The code contains **1** high severity issue. The security score is **5** out of **10**.

Summary

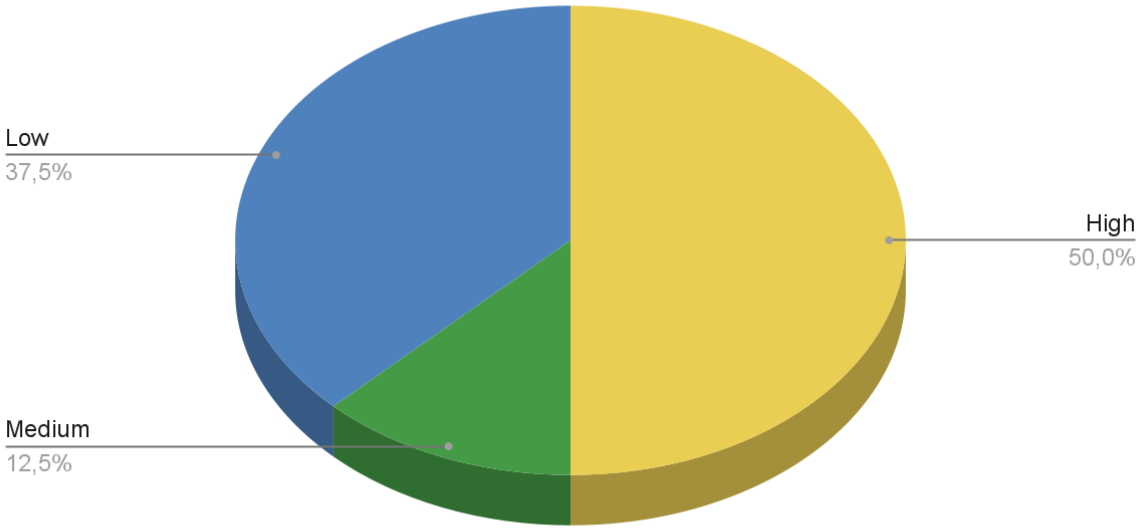
According to the assessment, the Customer's smart contract has the following score: **6.3**

Notices

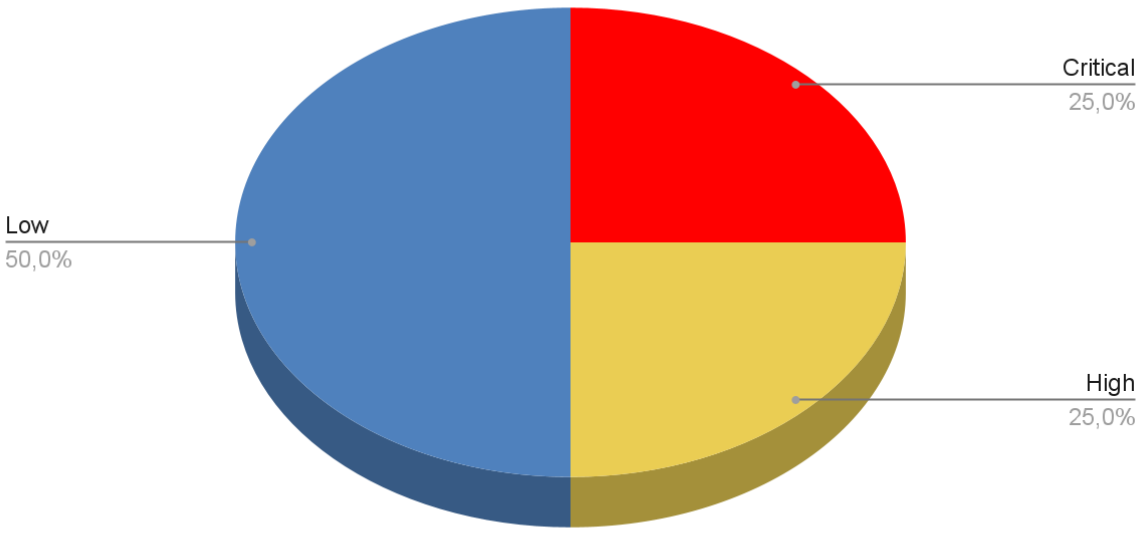
1. There are other smart contracts in the repository that are not included in the audit scope.
2. The MyLiquidityPool contract is “UsingLiquidityProtectionService”, specifying the address of its deployed version, the code of which cannot be viewed.

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

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

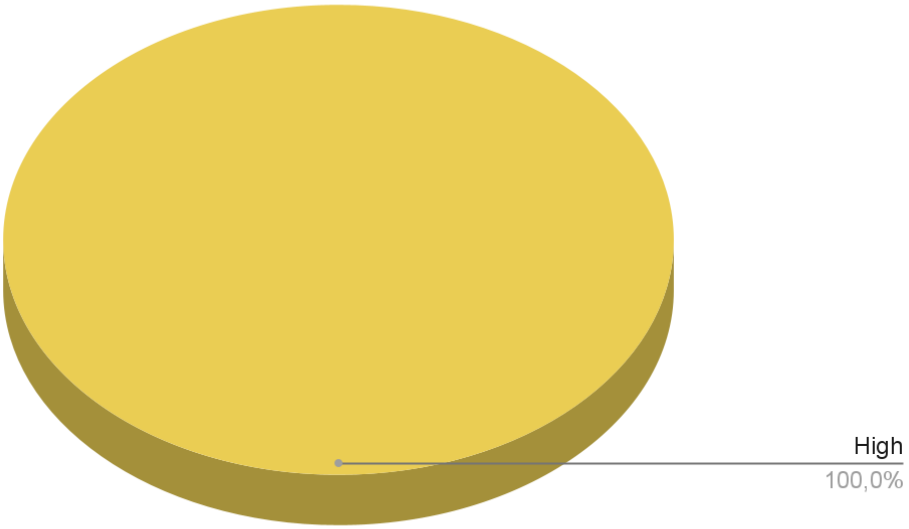


Graph 2. The distribution of vulnerabilities after the second review.



Graph 3. The distribution of vulnerabilities after the third review.

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



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 cannot lead to assets loss or data manipulations.
Low	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that cannot have a significant impact on execution

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

Findings

Critical

1. **Incorrect token amount calculation in swap.**

   The amount of MLP token (“mlpAmount”) in swap is calculated by the addition of amount and exchange rate (_amount + ratesPrecision / rate) instead of their multiplication (_amount * ratesPrecision / rate).

   Therefore, the amount of MLP token will not be calculated correctly.

   Contracts: PrivateRound.sol

   Function: swap

   Recommendation: perform the calculation of the amount of MLP token by the multiplication of the amount and exchange rate.

   Status: Fixed (Revised Commit: 489d591df02c0307ce4b9312460fd2dad44637fe)

High

1. **Owner`s ability to change user information.**

   The owner can change information about the amount of tokens the user has exchanged and claimed.

   This may lead to manipulations of the amount of tokens the user has to claim.

   Contracts: PrivateRound.sol

   Function: updateUserInfo

   Recommendation: deprive the owner of the right to change the user's information.

   Status: Fixed (Revised Commit: 283af42b62dbd5e5620251063b26b3f7e28eab13)

2. **Owner`s ability to change tokens and rate anytime.**

   The owner can change tokens addresses and swap rate anytime.

   Contracts: PublicRound.sol

   Function: updateUSDTToken, updateMLPToken, updateRate

   Recommendation: make a two-step conversion tokens change, where the first step is a request for a change, and the second is its acceptance after a certain time or number of blocks. Do the same for rate or validate the amount of tokens that users expect to match the one they receive.

   Status: Fixed (Revised Commit: 283af42b62dbd5e5620251063b26b3f7e28eab13)

3. **No whitelists in token selling.**

   According to the documentation, contracts are KYC Secured, but there are no whitelists in the code.

   Contracts: PrivateRound.sol

   Function: -

   Recommendation: add whitelists to the contracts.

   Status: **Mitigated. The Customer approved that whitelists are not implied.**

4. **No guaranteed tokens in the contract.**

   No tokens are guaranteed in the contract, so there is no certainty that users will receive any tokens.

   Contracts: PrivateRound.sol

   Function: -

   Recommendation: ensure the contract balance is sufficient before accepting the user's funds.

   Status: **Fixed (Revised Commit: 283af42b62dbd5e5620251063b26b3f7e28eab13)**

5. **Highly permissive owner access.**

   The functionality allows the owner to change users` “amount” and “claimed” values.

   This may lead to manipulations of the amount of tokens the user has to claim. Such behaviour is not described in the documentation.

   Contracts: PrivateRound.sol

   Function: migrateUsers

   Recommendation: deprive the owner of the right to change the user's information.

   Status: **Acknowledged**

Medium

1. **Using SafeMath.**

   Both SafeMath and ordinary mathematical operations are used. SafeMath is generally not needed starting with Solidity 0.8.

   Contracts: PrivateRound.sol

   Function: -

   Recommendation: remove SafeMath.

   Status: **Fixed (Revised Commit: 283af42b62dbd5e5620251063b26b3f7e28eab13)**

Low

1. **Unlocked pragma.**

   Contracts with unlocked pragmas may be deployed by the latest compiler, which may have higher risks of undiscovered bugs.

   Contracts: PublicRound.sol, PrivateRound.sol

   Recommendation: lock pragmas to a specific compiler version.

   Status: **Fixed (Revised Commit: 283af42b62dbd5e5620251063b26b3f7e28eab13)**

2. **No events on state variables changings.**

   It is recommended to emit events if the contract's state is changed.

   Contracts: PrivateRound.sol

   Functions: updateUserInfo

   Recommendation: emit the events on changing state variables.

   Status: **Fixed (Revised Commit: 283af42b62dbd5e5620251063b26b3f7e28eab13)**

3. **Hardcoded percentages and duration.**

   According to the documentation, there are different reward percentages and the duration for getting them (monthly or weekly). However, only one type of percentage (10% for the first period and 5% for all the next) and only "monthly" duration are hardcoded.

   Contracts: PrivateRound.sol

   Functions: -

   Recommendation: add the ability to customize this data.

   Status: **Fixed (Revised Commit: 283af42b62dbd5e5620251063b26b3f7e28eab13)**

4. **Unlocked pragma.**

   Contracts with unlocked pragmas may be deployed by the latest compiler, which may have higher risks of undiscovered bugs.

   Contracts: token.sol

   Recommendation: lock pragmas to a specific compiler version.

   Status: **Fixed (Revised Commit: 489d591df02c0307ce4b9312460fd2dad44637fe)**

5. **Using "public" visibility with constructor.**

   Constructors visibility (public or external) is not needed for constructors starting from Solidity v0.7.0.

   Contracts: token.sol

Functions: constructor

Recommendation: remove the “public” visibility from the constructor.

Status: **Fixed (Revised Commit: 489d591df02c0307ce4b9312460fd2dad44637fe)**

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 by the best industry practices at the date of this report, with 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 a sufficient assessment regarding the utility and safety of the code, bug-free status, or any other contract statements. 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 cannot guarantee the explicit security of the audited smart contracts.