HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: PolkaBridge
Date: January 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 PolkaBridge.
Approved by	Andrew Matiukhin | CTO Hacken OU
Type	ERC20 token; Staking; DEX
Platform	Ethereum / Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Repository	https://github.com/cyclese96/PolkaBridge-DEX
Commit	21b662c48caf08242bfa01621bbbafc957e4ff31
Technical Documentation	YES
JS tests	YES
Website	polkabridge.org
Timeline	14 DECEMBER 2021 - 26 JANUARY 2022
Changelog	23 DECEMBER 2021 - INITIAL AUDIT 26 JANUARY 2022 - 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 PolkaBridge (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 December 8th, 2021 - December 23rd, 2021.

The second review conducted on January 26th, 2022.

Scope

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

Repository:
https://github.com/cyclese96/PolkaBridge-DEX
Commit:
21b662c48caf08242bfa01621bbbafc957e4ff31
Technical Documentation: Yes, in the repository
JS tests: Yes, in the repository
Contracts:
factory/contracts/UniswapV2Factory.sol
factory/contracts/UniswapV2ERC20.sol
factory/contracts/libraries/SafeMath.sol
factory/contracts/libraries/UQ112x112.sol
factory/contracts/libraries/Math.sol
factory/contracts/utils/OwnableFactory.sol
factory/contracts/UniswapV2Pair.sol
factory/contracts/interfaces/IERC20.sol
factory/contracts/interfaces/IUniswapV2ERC20.sol
factory/contracts/interfaces/IUniswapV2Factory.sol
factory/contracts/interfaces/IUniswapV2Pair.sol
factory/contracts/interfaces/IUniswapV2Callee.sol
farming/Contracts/ReentrancyGuard.sol
farming/Contracts/PolkaBridgeFarm.sol
router/contracts/UniswapV2Migrator.sol
router/contracts/libraries/UniswapV2OracleLibrary.sol
router/contracts/libraries/UniswapV2Library.sol
router/contracts/libraries/SafeMath.sol
router/contracts/libraries/UniswapV2LiquidityMathLibrary.sol
router/contracts/UniswapV2Router02.sol
router/contracts/interfaces/V1/IUniswapV1Factory.sol
router/contracts/interfaces/V1/IUniswapV1Exchange.sol
router/contracts/interfaces/IERC20.sol
router/contracts/interfaces/IUniswapV2Router01.sol
router/contracts/interfaces/IUniswapV2ERC20.sol
router/contracts/interfaces/IUniswapV2Router02.sol
router/contracts/interfaces/IWETH.sol
router/contracts/interfaces/IUniswapV2Migrator.sol
router/contracts/interfaces/IUniswapV2Factory.sol
router/contracts/interfaces/IUniswapV2Pair.sol

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

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

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

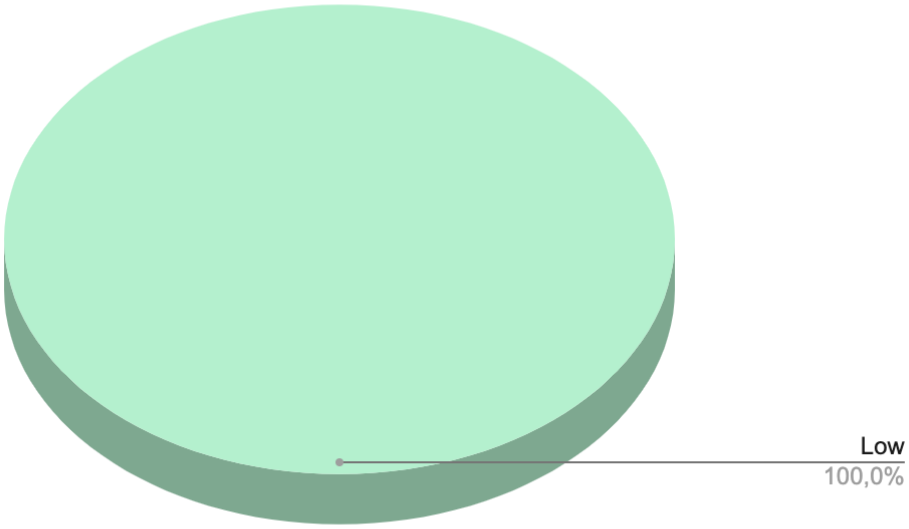
analysis were manually reviewed, and important vulnerabilities are presented in the Audit overview section. All found issues can be found in the Audit overview section.

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

After second review security engineers found **1** low severity issue.

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

Possible rewards lost or receive more

Contracts: PolkaBridgeFarm.sol

Changing *allocPoint* in the *set* method while *_withUpdate* flag set to *false* may lead to rewards lost or receiving rewards more than deserved.

Recommendation: Please call *updatePool(_pid)* in the case if *_withUpdate* flag is *false* and you don't want to update all pools.

Status: fixed

Medium

No medium severity issues were found.

Low

1. The function iterates over array of unpredictable size

   Contracts: PolkaBridgeFarm.sol

   Functions: massUpdatePools

   Gas consumption grows with array size and starting from a certain size function could become inoperable.

   Recommendation: limit *poolInfo[]* size

2. Missing event for changing *poolInfo[]*, *totalAllocPoint*, *migrator*

   Contracts: PolkaBridgeFarm.sol

   Functions: add, set, setMigrator

   Changing critical values should be followed by the event emitting for better tracking off-chain.

   Recommendation: Please emit events on the critical values changing

   Status: fixed

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

3. A public function that could be declared external.

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

   Contracts: PolkaBridgeFarm.sol, UniswapV2Router02.sol

   Functions: *add, set, setMigrator, migrate, deposit, withdraw, emergencyWithdraw, removeLiquidityETH*, quote, getAmountOut, getAmountIn

   Recommendation: Use the **external** attribute for functions never called from the contract.

   Status: fixed

4. Using SafeMath in Solidity >= 0.8.0

   Starting solidity version 0.8.0 arithmetic operations revert on underflow and overflow. There's no more need to assert the result of operations.

   Contracts: PolkaBridgeFarm.sol, UniswapV2Pair.sol, UniswapV2ERC20.sol

   Recommendation: Please avoid using assert for arithmetic operations.

   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 and **4** low severity issues.

After second review security engineers found **1** low severity issue.

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.